Neural Sign Language Translation

Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Herman Ney, Richard Bowden

1University of Surrey, {n.camgoz, s.hadfield, r.bowden}@surrey.ac.uk
2RWTH Aachen University, {koller, ney}@cs.rwth-aachen.de

Abstract

Sign Language Recognition (SLR) has been an active research field for the last two decades. However, most research to date has considered SLR as a naive gesture recognition problem. SLR seeks to recognize a sequence of continuous signs but neglects the underlying rich grammatical and linguistic structures of sign language that differ from spoken language. In contrast, we introduce the Sign Language Translation (SLT) problem. Here, the objective is to generate spoken language translations from sign language videos, taking into account the different word orders and grammar.

We formalize SLT in the framework of Neural Machine Translation (NMT) for both end-to-end and pretrained settings (using expert knowledge). This allows us to jointly learn the spatial representations, the underlying language model, and the mapping between sign and spoken language.

To evaluate the performance of Neural SLT, we collected the first publicly available Continuous SLT dataset, RWTH-PHOENIX-Weather 2014T. It provides spoken language translations and gloss level annotations for German Sign Language videos of weather broadcasts. Our dataset contains over 95M frames with >67K signs from a sign vocabulary of >1K and >99K words from a German vocabulary of >2.8K. We report quantitative and qualitative results for various SLT setups to underpin future research in this newly established field. The upper bound for translation performance is calculated at 19.26 BLEU-4, while our end-to-end frame-level and gloss-level tokenization networks were able to achieve 9.58 and 18.13 respectively.

1. Introduction

Sign Languages are the primary language of the deaf community. Despite common misconceptions, sign languages have their own specific linguistic rules [55] and do not translate the spoken languages word by word. Therefore, the numerous advances in SLR [15] and even the move to the challenging Continuous SLR (CSLR) [33, 36] problem, do not allow us to provide meaningful interpretations of what a signer is saying. This translation task is illustrated in Figure 1, where the sign language glosses give the meaning and the order of signs in the video, but the spoken language equivalent (which is what is actually desired) has both a different length and ordering.

Most of the research that has been conducted in SLR to date has approached the task as a basic gesture recognition problem, ignoring the linguistic properties of the sign language and assuming that there is a one-to-one mapping of sign to spoken words. Contrary to SLR, we propose to approach the full translation problem as a NMT task. We use state-of-the-art sequence-to-sequence (seq2seq) based deep learning methods to learn: the spatio-temporal representation of the signs, the relation between these signs (in other words the language model) and how these signs map to the spoken or written language. To achieve this we introduce new vision methods, which mirror the tokenization and embedding steps of standard NMT. We also present the first continuous SLT dataset, RWTH-PHOENIX-Weather 2014T, to allow future research to be conducted towards sign to spoken language translation. The contributions of this paper can be summarized as:

- The first exploration of the video to text SLT problem.
- The first publicly available continuous SLT dataset, PHOENIX14T, which contains video segments, gloss annotations and spoken language translations.
- A broad range of baseline results on the new corpus including a range of different tokenization and attention schemes in addition to parameter recommendations.

The rest of this paper is organized as follows: In Section 2 we survey the fields of sign language recognition, seq2seq learning and neural machine translation. In Section 3 we formalize the SLT task in the framework of neural machine translation and describe our pipeline. We then intro-
duce RWTH-PHOENIX-Weather 2014T, the first continuous SLT dataset, in Section 4. We share our quantitative and qualitative experimental results in Sections 5 and 6, respectively. Finally, we conclude our paper in Section 7 by discussing our findings and the future of the field.

2. Related Work

There are various factors that have hindered progress towards SLT. Although there have been studies such as [9], which recognized isolated signs to construct sentences, to the best of our knowledge no dataset or study exists that achieved SLT directly from videos, until now. In addition, existing linguistic work on SLT has solely dealt with text to text translation. Despite only including textual information, these have been very limited in size (averaging 3000 total words) [46, 54, 52]. The first important factor is that collection and annotation of continuous sign language data is a laborious task. Although there are datasets available from linguistic sources [51, 28] and sign language interpretations from broadcasts [14], they are weakly annotated and lack the human pose information which legacy sign language recognition methods heavily relied on. This has resulted in many researchers collecting isolated sign language datasets [63, 7] in controlled environments with limited vocabulary, thus inhibiting the end goal of SLT. The lack of a baseline dataset for SLT has rendered most research incomparable, robbing the field of competitive progress.

With the development of algorithms that were capable of learning from weakly annotated data [5, 50, 14] and the improvements in the field of human pose estimation [10, 59, 8], working on linguistic data and sign language interpretations from broadcasts became a feasible option. Following these developments, Forster et al. released RWTH-PHOENIX-Weather 2012 [20] and its extended version RWTH-PHOENIX-Weather 2014 [21], which was captured from sign language interpretations of weather forecasts. The PHOENIX datasets were created for CSLR and they provide sequence level gloss annotations. These datasets quickly became a baseline for CSLR.

Concurrently, Deep Learning (DL) [59] has gained popularity and achieved state-of-the-art performance in various fields such as Computer Vision [38], Speech Recognition [2] and more recently in the field of Machine Translation [47]. Until recently SLR methods have mainly used hand-crafted intermediate representations [53, 16] and the temporal changes in these features have been modelled using classical graph based approaches, such as Hidden Markov Models (HMMs) [58], Conditional Random Fields [62] or template based methods [5, 38]. However, with the emergence of DL, SLR researchers have quickly adopted Convolutional Neural Networks (CNNs) [40] for manual [35, 37] and non-manual [54] feature representation, and Recurrent Neural Networks (RNNs) for temporal modelling [6, 36, 17].

One of the most important breakthroughs in DL was the development of seq2seq learning approaches. Strong annotations are hard to obtain for seq2seq tasks, in which the objective is to learn a mapping between two sequences. To be able to train from weakly annotated data in an end-to-end manner, Graves et al. proposed Connectionist Temporal Classification (CTC) Loss [25], which considers all possible alignments between two sequences while calculating the error. CTC quickly became a popular loss layer for many seq2seq applications. It has obtained state-of-the-art performance on several tasks in speech recognition [27, 2] and clearly dominates hand writing recognition [25]. Computer vision researchers adopted CTC and applied it to weakly labeled visual problems, such as lip reading [3], action recognition [30], hand shape recognition [6] and CSLR [6, 17].

Another common seq2seq task is machine translation, which aims to develop methods that can learn the mapping between two languages. Although CTC is popular, it is not suitable for machine translation as it assumes source and target sequences share the same order. Furthermore, CTC assumes conditional independence within target sequences, which doesn’t allow networks to learn an implicit language model. This led to the development of Encoder-Decoder Network architectures [31] and the emergence of the NMT field [47]. The main idea behind Encoder-Decoder Networks is to use an intermediary latent space to map two sequences, much like the latent space in auto-encoders [24], but applied to temporal sequences. This is done by first encoding source sequences to a fixed sized vector and then decoding target sequences from this. The first architecture proposed by Kalchbrenner and Blunsom [31] used a single RNN for both encoding and decoding tasks. Later Sutskever et al. [56] and Cho et al. [11] proposed delegating encoding and decoding to two separate RNNs.

Although encoder-decoder networks improved machine translation performance, there is still the issue of an information bottleneck caused by encoding the source sequence into a fixed sized vector and the long term dependencies between source and target sequence. To address these issues, Bahdanau et al. [4] proposed passing additional information to the decoder using an attention mechanism. Given encoder outputs, their attention function calculates the alignment between source and target sequences. Luong et al. [44] further improved this approach by introducing additional types of attention score calculation and the input-feeding approach. Since then, various attention based architectures have been proposed for NMT, such as GNMT [60] which combines bi-directional and uni-directional encoders in a deep architecture and [22] which introduced a convolution based seq2seq learning approach. Similar attention based approaches have been applied to various Computer Vision tasks, such as image captioning [61], lip reading [13] and action recognition [19].
3. Neural Sign Language Translation

Translating sign videos to spoken language is a seq2seq learning problem by nature. Our objective is to learn the conditional probability \( p(y|x) \) of generating a spoken language sentence \( y = (y_1, y_2, \ldots, y_U) \) with \( U \) number of words given a sign video \( x = (x_1, x_2, \ldots, x_T) \) with \( T \) number of frames. This is not a straight forward task as the number of frames in a sign video is much higher than the number of words in its spoken language translation (i.e. \( T \gg U \)). Furthermore, the alignment between sign and spoken language sequences are usually unknown and non-monotonic. In addition, unlike other translation tasks that work on text, our source sequences are videos. This renders the use of classic sequence modeling architectures such as the RNN difficult. Instead, we propose combining CNNs with attention-based encoder-decoders to model the conditional probability \( p(y|x) \). We experiment with training our approach in an end-to-end manner to jointly learn the alignment and the translation of sign language videos to spoken language sentences. An overview of our approach can be seen in Figure 2.

In the remainder of this section, we will describe each component of our architecture in detail.

3.1. Spatial and Word Embeddings:

Neural machine translation methods start with tokenization of source and target sequences and projecting them to a continuous space by using word embeddings [45]. The main idea behind using word embeddings is to transform the sparse one-hot vector representations, where each word is equidistant from each other, into a denser form, where words with similar meanings are closer. These embeddings are either learned from scratch or pretrained on larger datasets and fine-tuned during training. However, contrary to text, signs are visual. Therefore, in addition to using word embeddings for our target sequences (spoken language sentences), we need to learn spatial embeddings to represent sign videos. To achieve this we utilize 2D CNNs. Given a sign video \( x \), our CNN learns to extract non-linear frame level spatial representations as:

\[
f_t = \text{SpatialEmbedding}(x_t) \tag{1}
\]

where \( f_t \) corresponds to the feature vector produced by propagating a video frame \( x_t \) through our CNN.

For word embedding, we use a fully connected layer that learns a linear projection from one-hot vectors of spoken language words to a denser space as:

\[
g_u = \text{WordEmbedding}(y_u) \tag{2}
\]

where \( g_u \) is the embedded version of the spoken word \( y_u \).

3.2. Tokenization Layer:

In NMT the input and output sequences can be tokenized at many different levels of complexity: characters, words, N-grams or phrases. Low level tokenization schemes, such as the character level, allow smaller vocabularies to be used, but greatly increase the complexity of the sequence modeling problem, and require long term relationships to be maintained. High level tokenization makes the recognition problem far more difficult due to vastly increased vocabularies, but the language modeling generally only needs to consider a small number of neighboring tokens.

As there has been no previous research on SLT, it is not clear what tokenization schemes are most appropriate for this problem. This is exacerbated by the fact that, unlike NMT research, there is no simple equivalence between the tokenizations of the input sign video and the output text. The framework developed in this paper is generic and can use various tokenization schemes on the spatial embeddings sequence \( f_{1:T} \)

\[
z_{1:N} = \text{Tokenization}(f_{1:T}) \tag{3}
\]
3.3. Attention-based Encoder-Decoder Networks:

To be able to generate the target sentence $y$ from tokenized embeddings $z_{1:N}$ of a sign video $x$, we need to learn a mapping function $B(z_{1:N}) \rightarrow y$ which will maximize the probability $p(y|x)$. We propose modelling $B$ using an attention-based encoder-decoder network, which is composed of two specialized deep RNNs. By using these RNNs we break down the task into two phases. In the encoding phase, a sign videos’ features are projected into a latent space in the form of a fixed size vector, later to be used in the decoding phase for generating spoken sentences.

During the encoding phase, the encoder network, reads in the feature vectors one by one. Given a sequence of representations $z_{1:N}$, we first reverse its order in the temporal domain, as suggested by [56], to shorten the long term dependencies between the beginning of sign videos and spoken language sentences. We then feed the reversed sequence $z_{N:1}$ to the Encoder which models the temporal changes in video frames and compresses their cumulative representation in its hidden states as:

$$o_n = \text{Encoder}(z_n, o_{n+1})$$

(4)

where $o_n$ is the hidden state produced by recurrent unit $n$, $o_{N+1}$ is a zero vector and the final encoder output $o_1$ corresponds to the latent embedding of the sequence $h_{\text{sign}}$ which is passed to the decoder.

The decoding phase starts by initializing hidden states of the decoder network using the latent vector $h_{\text{sign}}$. In the classic encoder-decoder architecture [56], this latent representation is the only information source of the decoding phase. By taking its previous hidden state ($h_{u-1}$) and the word embedding ($g_{u-1}$) of the previously predicted word ($y_{u-1}$) as inputs, the decoder learns to generate the next word in the sequence ($y_u$) and update its hidden state ($h_u$):

$$y_u, h_u = \text{Decoder}(g_{u-1}, h_{u-1})$$

(5)

where $h_0 = h_{\text{sign}}$ is the spatio-temporal representation of sign language video learned by the Encoder and $y_0$ is the special token $<$ bos $>$ indicating the beginning of a sentence. This procedure continues until another special token $<$ eos $>$, which indicates the end of a sentence, is predicted. By generating sentences word by word, the Decoder decomposes the conditional probability $p(y|x)$ into ordered conditional probabilities:

$$p(y|x) = \prod_{u=1}^{U} p(y_u|y_{1:u-1}, h_{\text{sign}})$$

(6)

which is used to calculate the errors by applying cross entropy loss for each word. For the end-to-end experiments, these errors are back propagated through the encoder-decoder network to the CNN and word embeddings, thus updating all of the network parameters.

Attention Mechanisms:

A major drawback of using a classic encoder-decoder architecture is the information bottleneck caused by representing a whole sign language video with a fixed sized vector. Furthermore, due to large number of frames, our networks suffer from long term dependencies and vanishing gradients. To overcome these issues, we utilize attention mechanisms to provide additional information to the decoding phase. By using attention mechanisms our networks are able to learn where to focus while generating each word, thus providing the alignment of sign videos and spoken language sentences. We employ the most prominent attention approach proposed by Bahdanau et al. [4] and later improved by Luong et al. [44].

The main idea behind attention mechanisms is to create a weighted summary of the source sequence to aid the decoding phase. This summary is commonly known as the context vector and it will be noted as $c_u$ in this paper. For each decoding step $u$, a new context vector $c_u$ is calculated by taking a weighted sum of encoder outputs $o_{1:N}$ as:

$$c_u = \sum_{n=1}^{N} \gamma_n^u o_n$$

(7)

where $\gamma_n^u$ represent the attention weights, which can be interpreted as the relevance of an encoder input $z_n$ to generating the word $y_u$. When visualized, attention weights also help to display the alignments between sign videos and spoken language sentences learned by the encoder-decoder network. These weights are calculated by comparing the decoder hidden state $h_u$ against each output $o_n$ as:

$$\gamma_n^u = \frac{\exp(score(h_u, o_n))}{\sum_{n'=1}^{N} \exp(score(h_u, o_{n'}))}$$

(8)

where the scoring function depends on the attention mechanism that is being used. In this work we examine two scoring functions. The first one is a multiplication based approach proposed by Luong et al. [44] and the second is a concatenation based function proposed by Bahdanau et al. [4]. These functions are as follows:

$$\text{score}(h_u, o_n) = \begin{cases} h_u^\top W o_n & \text{[Multiplication]} \\ V^\top \tanh(W[h_u; o_n]) & \text{[Concatenation]} \end{cases}$$

(9)

where $W$ and $V$ are learned parameters. The context vector $c_u$ is then combined with the hidden state $h_u$ to calculate the attention vector $a_u$ as:

$$a_u = \tanh(W_c[c_u; h_u])$$

(10)

Finally, we feed the $a_u$ to a fully connected layer to model the ordered conditional probability in Equation[6]. Furthermore $a_{u-1}$ is fed to the next decoding step $u+1$ thus changing Equation[5] to:

$$y_u, h_u = \text{Decoder}(g_{u-1}, h_{u-1}, a_{u-1})$$

(11)
4. Sign Language Translation Dataset

As discussed in Section 2, there are no suitable datasets available to support research towards SLT. Due to the cost of annotation, existing linguistic datasets are too small to support deep learning.

In this work we present “RWTH-PHOENIX-Weather 2014T”, a large vocabulary, continuous SLT corpus. PHOENIX14T is an extension of the PHOENIX14 corpus, which has become the primary benchmark for SLR in recent years. PHOENIX14T constitutes a parallel corpus including sign language videos, sign-gloss annotations and also German translations (spoken by the news anchor), which are all segmented into parallel sentences. Due to different sentence segmentation between spoken language and sign language, it was not sufficient to simply add a spoken language tier to PHOENIX14. Instead, the segmentation boundaries also had to be redefined. Wherever the addition of a translation layer necessitated new sentence boundaries, we used the forced alignment approach of [35] to compute the new boundaries.

In addition to changes in boundaries, RWTH-PHOENIX-Weather 2014T has a marginally decreased vocabulary due to some improvements in the normalization schemes. This means performance on PHOENIX14 and PHOENIX14T will be similar, but not exactly comparable. However, care has been taken to assure that the dev/test sets of PHOENIX14 do not overlap with the new PHOENIX14T training set and also that none of the new dev/test sets from PHOENIX14T overlap with the PHOENIX14 training set.

This corpus is publicly available to the research community for facilitating the future growth of SLT research. The detailed statistics of the dataset can be seen in Table 1. OOV stands for Out-Of-Vocabulary, e.g. words that occur in test, but not in training. Singletons occur only once in the training set. The corpus covers unconstrained sign language of 9 different signers with a vocabulary of 1066 different signs and translations into German spoken language with a vocabulary of 2887 different words. The corpus features professional sign language interpreters and has been annotated using sign glosses by deaf specialists. The spoken German translation originates from the news speaker. It has been automatically transcribed, manually verified and normalized.

<table>
<thead>
<tr>
<th>Sign Gloss</th>
<th>German</th>
</tr>
</thead>
<tbody>
<tr>
<td>segments</td>
<td>7,096</td>
</tr>
<tr>
<td>frames</td>
<td>827,354</td>
</tr>
<tr>
<td>vocab.</td>
<td>1,066</td>
</tr>
<tr>
<td>tot. words</td>
<td>67,781</td>
</tr>
<tr>
<td>tot. OOVs</td>
<td>-</td>
</tr>
<tr>
<td>singletons</td>
<td>337</td>
</tr>
</tbody>
</table>

Table 1. Key statistics of the new dataset.

5. Quantitative Experiments

Using our new PHOENIX14 dataset, we conduct several sets of experiments to create a baseline for SLT. We categorize our experiments under three groups:

1. Gloss2Text (G2T), in which we simulate having a perfect SLR system as an intermediate tokenization.
2. Sign2Text (S2T) which covers the end-to-end pipeline translating directly from frame level sign language video into spoken language.
3. Sign2Gloss2Text (SG2T) which uses a SLR system as tokenization layer to add intermediate supervision.

All of our encoder-decoder networks were built using four stacked layers of residual recurrent units with separate parameters. Each recurrent layer contains 1000 hidden units. In our S2T experiments we use AlexNet without its final layer (fc8) as our Spatial Embedding Layer and initialize it using weights that were trained on ImageNet [18]. For our SG2T experiments we use the CNN-RNN-HMM network proposed by Koller et al. [36] as our Tokenization Layer, which is the state-of-the-art CSLR. It achieves a gloss recognition performance of 25.7%/26.6% word error rate on the dev/test sets of the PHOENIX14T. All remaining parts of our networks are initialized using Xavier [23] initialization. We use Adam [32] optimization method with a learning rate of $10^{-5}$ and its default parameters. We also use gradient clipping with a threshold of 5 and dropout connections with a drop probability of 0.2.

All of our networks are trained until the training perplexity is converged, which took $\sim 30$ epochs on average. We evaluate our models on dev/test sets every half-epoch, and report results for each setup using the model that performed the best on the dev set. In the decoding phase we generate spoken language sentences using beam search with a beam width of three, which we empirically shows to be the optimal beam size.

To measure our translation performance we utilize BLEU [49] and ROGUE [42] scores, which are commonly used metrics for machine translation. As ROUGE score we use ROUGE-L F1-Score, while as BLEU score we report BLEU-1,2,3,4 to give a better perspective of the translation performance on different phrase levels.

We implemented our networks using TensorFlow [1]. Our code, which is based on Luong et al.’s NMT library [43], is made publicly available.

5.1. G2T: Simulating Perfect Recognition

Our SLT framework supports various input tokenizations. In our first set of experiments we simulate using an idealized SLR system as an intermediate tokenizer. NMT networks are trained to generate spoken language translations from ground truth sign glosses. We refer to this as G2T.

[https://github.com/neccam/nslt]
There are two main objectives of the G2T experiments. First to create an upper bound for end-to-end SLT. Second to examine different encoder-decoder network architectures and hyper-parameters, and evaluate their effects on sign to spoken language translation performance. As training S2T networks is an order of magnitude slower than G2T, we use the best setup from our G2T experiments when training our S2T networks.

Note that we should expect the translation performance’s upper bound to be significantly lower than 100%. As in all natural language problems, there are many ways to say the same thing, and thus many equally valid translations. Unfortunately, this is impossible to quantify using any existing evaluation measure.

### 5.1.1 Recurrent Units: GRUs vs LSTMs

Various types of recurrent units have been used for neural machine translation. The first encoder-decoder network proposed by Kalchbrenner and Blunsom [31] was built using a single RNN with vanilla recurrent units. Later approaches employed shallow [50, 44] and deep architectures [60] of Long Short-Term Memory (LSTM) units [29] and Gated Recurrent Units (GRUs) [12]. To choose which recurrent unit to use, our first experiment trained two G2T networks using LSTMs and GRUs. Both networks were trained using a batch size of 128 and Luong attention which were described in detail in Section 3. We also trained a network which did not use any attention mechanisms. All of our networks were trained using Gated Recurrent Units and a batch size of 128.

Our first observation from this experiment was that having an attention mechanism improved the translation performance drastically as shown in Table 3. When attention mechanisms are compared, Luong attention outperformed Bahdanau attention and generalized better to the test set. We believe this is due to Luong attention’s use of the decoder network’s hidden state at time \( u \) while generating the target word \( w_{u} \). We train our remaining G2T networks using Luong attention.

#### 5.1.3 What Batch Size to use?

There have been several studies on the effects of batch sizes while using Stochastic Gradient Descent (SGD) [41]. Although larger batch sizes have the advantage of providing smoother gradients, they decrease the rate of convergence. Furthermore, recent studies on the information theory behind deep learning suggests the noise provided by smaller batch size helps the networks to represent the data more efficiently [57, 53]. In addition, training and evaluation set distributions of seq2seq datasets are distinct by nature. When early stopping is employed during training, having additional noise provided by smaller batch sizes gives the optimization the opportunity to step closer to the target distribution. This suggests there is an optimal batch size given a network setup. Therefore, in our third set of experiments we evaluate the effects of the batch size on the translation. We train five G2T networks using different batch sizes that are 128, 64, 32, 16 and 1. All of our networks were trained using GRUs and Luong attention.

One interesting observation from this experiment was that, the networks trained using smaller batch sizes converged faster but to a higher training perplexity than one. We believe this is due to high variance between gradients. To deal with this we decrease the learning rate to \( 10^{-6} \).
when the training perplexity plateau, and continue training for 100,000 iterations. Results show that having a smaller batch size helps the translation performance. As reported in Table 4, the G2T network with batch size one outperformed networks that were trained using larger batch sizes. Considering these results, the remainder of our experiments use a batch size of one.

### 5.1.4 Effects of Beam Width

The most straight forward decoding approach for Encoder-Decoder networks is to use a greedy search, in which the word with highest probability is considered the prediction and fed to the next time step of the decoder. However, this greedy approach is prone to errors, given that the predictions can have low confidence. To address this, we use a simple left-to-right Beam Search during the decoding phase, in which a number of candidate sequences, also known as beam width, are stored and propagated through the decoder. However, larger beam width does not necessarily mean better translation performance and increases decoding duration and memory requirements. Therefore, to find the optimal value, we use our best performing G2T network to do a parameter search over possible beam widths and report development and test set translation performances in the form of a BLEU-4 score.

![Figure 3. Effects of Beam Width on G2T performance.](image)

As shown in Figure 3, a beam width of two or three was optimal for our G2T network. Although beam width two yielded the highest translation performance on the development set, beam width three generalized better to the test set. In addition, as beam width increased, the BLEU-4 scores plateau and then start to decline. Taking these results into consideration, we continue using beam width three for the rest of our experiments.

### 5.2. S2T: From Sign Video To Spoken Text

In our second set of experiment we evaluate our S2T networks which learns to generate spoken language from sign videos without any intermediate representation in an end-to-end manner. In this setup our tokenization layer is an Identity function, feeding the spatial embeddings directly to the encoder-decoder network. Using the hyper-parameters from our G2T experiments, we train three S2T networks with different attention choices.

As with the G2T task, utilizing attention mechanisms increases the translation performance of our S2T networks (See Table 5). However, when compared against G2T, the translation performance is lower. We believe this might due to several reasons. As the number of frames in a sign video is much higher than the number of its gloss level representations, our S2T networks suffer from long term dependencies and vanishing gradients. In addition, the dataset we are using might be too small to allow our S2T network to generalize considering the number of parameters (CNN+EncoderDecoder+Attention). Furthermore, expecting our networks to recognize visual sign languages and translate them to spoken languages with single supervision might be too much to ask from them. Therefore in our next set of experiments, which we call S2G2T, we introduce the gloss level supervision to aid the task of full translation from sign language videos.

### 5.3. S2G2T: Gloss Level Supervision

In our final experiment we propose using glosses as an intermediate representation while going from sign videos to spoken videos without any retraining to generate sentences from the estimated gloss token embeddings. In the second setup: Sign2Gloss→Gloss2Text (S2G→G2T), we use our best performing G2T network without any retraining to generate sentences from the estimated gloss token embeddings. In the second setup: S2G→G2T, we train a network from scratch to learn to translate from the predicted gloss.

The S2G→G2T network performs surprisingly well considering there was no additional training. This shows us that

---

**Table 5. S2T: Attention Mechanism Experiments.**

<table>
<thead>
<tr>
<th>Attention:</th>
<th>ROUGE</th>
<th>BLEU-1</th>
<th>BLEU-2</th>
<th>BLEU-3</th>
<th>BLEU-4</th>
<th>ROUGE</th>
<th>BLEU-1</th>
<th>BLEU-2</th>
<th>BLEU-3</th>
<th>BLEU-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>31.00</td>
<td>28.10</td>
<td>16.81</td>
<td>11.82</td>
<td>9.12</td>
<td>29.70</td>
<td>27.10</td>
<td>15.61</td>
<td>10.82</td>
<td>8.35</td>
</tr>
<tr>
<td>Bahdanau</td>
<td>31.80</td>
<td>31.87</td>
<td>19.11</td>
<td>13.16</td>
<td>9.94</td>
<td>31.80</td>
<td>32.24</td>
<td>19.03</td>
<td>12.83</td>
<td>9.58</td>
</tr>
<tr>
<td>Luong</td>
<td>32.6</td>
<td>31.58</td>
<td>18.98</td>
<td>13.22</td>
<td>10.00</td>
<td>30.70</td>
<td>29.86</td>
<td>17.52</td>
<td>11.96</td>
<td>9.00</td>
</tr>
</tbody>
</table>

**Table 6. Effects of different tokenization schemes for sign to text translation.**

<table>
<thead>
<tr>
<th>Approach:</th>
<th>ROUGE</th>
<th>BLEU-1</th>
<th>BLEU-2</th>
<th>BLEU-3</th>
<th>BLEU-4</th>
<th>ROUGE</th>
<th>BLEU-1</th>
<th>BLEU-2</th>
<th>BLEU-3</th>
<th>BLEU-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>G2T</td>
<td>46.02</td>
<td>44.40</td>
<td>31.83</td>
<td>24.61</td>
<td>20.16</td>
<td>45.45</td>
<td>44.13</td>
<td>31.47</td>
<td>23.89</td>
<td>19.26</td>
</tr>
<tr>
<td>S2T</td>
<td>31.80</td>
<td>31.87</td>
<td>19.11</td>
<td>13.16</td>
<td>9.94</td>
<td>31.80</td>
<td>32.24</td>
<td>19.03</td>
<td>12.83</td>
<td>9.58</td>
</tr>
<tr>
<td>S2G→G2T</td>
<td>43.76</td>
<td>41.08</td>
<td>29.10</td>
<td>22.16</td>
<td>17.86</td>
<td>43.45</td>
<td>41.54</td>
<td>29.52</td>
<td>22.24</td>
<td>17.79</td>
</tr>
<tr>
<td>S2G2T</td>
<td>44.14</td>
<td>42.88</td>
<td>30.30</td>
<td>23.02</td>
<td>18.40</td>
<td>43.80</td>
<td>43.29</td>
<td>30.39</td>
<td>22.82</td>
<td>18.13</td>
</tr>
</tbody>
</table>
our G2T network has already learned some robustness to noisy inputs, despite being trained on perfect glosses, this may be due to the dropout regularization employed during training. Our second approach S2G2T surpasses these results and obtains scores close to the idealized performance of the G2T network. This is likely because the translation system is able to correct the failure modes in the tokenizer. As can be seen in Table 6, compared to the S2T network S2G2T was able to surpass its performance by a large margin, indicating the importance of intermediary expert gloss level supervision to simplify the training process.

6. Qualitative Experiments

In this section we share our qualitative results. One of the most obvious ways of qualifying translation is to examine the resultant translations. To give a better understanding to the reader, in Table 7 we share translation samples generated from our G2T, S2T and S2G2T networks accompanied by the ground truth German and word to word English translations.

<table>
<thead>
<tr>
<th>Translation from our networks. (GT: Ground Truth)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GT: und nun die wettervorhersage für morgen samstag den zweiten april . (on sunday weather forecast for tomorrow saturday the second april)</td>
</tr>
<tr>
<td>S2T: und nun die wettervorhersage für morgen on sunday the second april . (and now the weatherforecast for tomorrow sunday the second april)</td>
</tr>
<tr>
<td>S2G2T: und nun die wettervorhersage für morgen on sunday the second april . (and now the weatherforecast for tomorrow sunny the twenty-second march)</td>
</tr>
<tr>
<td>GT: in der südwesthälfte regnet es zeitweise sonst ist es recht freundlich . (in the southwestpart it rains temporarily otherwise it is quite friendly)</td>
</tr>
<tr>
<td>S2T: in der südwesthälfte regnet es zeitleweise sonst ist es recht freundlich . (in the southwestern part it rains temporarily otherwise it is quite friendly)</td>
</tr>
<tr>
<td>S2G2T: von der südwesthälfte regnet es zeitweise sonst ist es recht freundlich . (from the southwestern it starts in many places)</td>
</tr>
<tr>
<td>GT: die neue woche beginnt wechselhaft und wieder kühl . (the new week starts unpredictable and again cooler)</td>
</tr>
<tr>
<td>S2T: die neue woche beginnt wechselhaft und wieder kühl . (the new week starts unpredictable and again cooler)</td>
</tr>
<tr>
<td>S2G2T: die neue woche beginnt wechselhaft und wieder kühl . (the new week starts unpredictable and again cooler)</td>
</tr>
<tr>
<td>GT: die neue woche beginnt wechselhaft und wieder kühl . (the new week starts unpredictable and again cooler)</td>
</tr>
<tr>
<td>S2T: von der südwesthälfte regnet es zeitweise sonst ist es recht freundlich . (in the southwestpart it rains temporarily otherwise it is quite friendly)</td>
</tr>
<tr>
<td>S2G2T: die neue woche beginnt wechselhaft und wieder kühl . (the new week starts unpredictable and again cooler)</td>
</tr>
<tr>
<td>GT: von der südwesthälfte regnet es zeitweise sonst ist es recht freundlich . (in the southwestpart it rains temporarily otherwise it is quite friendly)</td>
</tr>
<tr>
<td>S2T: die neue woche beginnt wechselhaft und wieder kühl . (the new week starts unpredictable and again cooler)</td>
</tr>
<tr>
<td>S2G2T: die neue woche beginnt wechselhaft und wieder kühl . (the new week starts unpredictable and again cooler)</td>
</tr>
</tbody>
</table>

We can see that the most common error mode is the mis-translation of dates, places and numbers. Although this does not effect the overall structure of the translated sentence, it tells us the embedding learned for these infrequent words could use some improvement.

In Figure 4 example attention maps can be seen for both the S2T and S2G2T systems. These maps show how dependent each output token (the horizontal axis) is on each input token (the vertical axis). The S2T network’s focus is concentrated primarily at the start of the video, but attention does jump to the end during the final words of the transla-

7. Conclusion

In this paper, we introduced the challenging task of Sign Language Translation and proposed the first end-to-end solution. In contrast to previous research, we took a machine translation perspective; treating sign language as a fully independent language and proposing SLT rather than SLR as the true route to facilitate communication with the deaf. To achieve NMT from sign videos, we employed CNN based spatial embedding, various tokenization methods including state-of-the-art RNN-HMM hybrids and attention-based encoder-decoder networks, to jointly learn to align, recognize and translate sign videos to spoken text.

To evaluate our approach we collected the first continuous sign language translation dataset, PHOENIX14T, which is publicly available. We conducted extensive experiments, making a number of recommendations to underpin future research.

As future work, it would be interesting to extend the attention mechanisms to the spatial domain to align building blocks of signs, also known as subunits, with their spoken language translations. It may also be possible to use an approach similar to SubUNets to inject specialist intermediate subunit knowledge, bridging the gap between S2T and S2G2T.

Acknowledgement

This work was funded by the SNSF Sinergia project “Scalable Multimodal Sign Language Technology for Sign Language Learning and Assessment (SMILE)” grant agreement number CRSII2_160811 and the European Union’s Horizon2020 research and innovation programme under grant agreement no. 762021 (Content4All). We would also like to thank NVIDIA Corporation for their GPU grant.
References


