Improving Sign Language Gloss Translation with Low-Resource Machine Translation Techniques

Xuan Zhang and Kevin Duh

Abstract A cascaded Sign Language Translation system first maps sign videos to gloss annotations and then translates glosses into spoken language text. This chapter focuses on the second-stage, gloss translation, which is challenging due to the scarcity of publicly available parallel data. We approach gloss translation as a low-resource machine translation task and investigate several popular methods for improving translation quality, including hyperparameter search, pretrained multilingual models, rule-based data augmentation, back-translation, and curriculum learning. We discuss the potentials and pitfalls of these methods based on experiments conducted on a German sign language dataset, RWTH-PHOENIX-Weather 2014T, and a Chinese sign language dataset, CSL daily. We also show how a sign-to-text translation system would benefit from the initialization of gloss-to-text checkpoints obtained with different methods. We further conduct word-level error analyses to study how the two components of the sign translation system contribute to the translation errors respectively.

1 Introduction

Sign language machine translation (SLMT) systems can be either an end-to-end system that maps sign language videos directly to spoken languages, or a cascaded system as shown in Fig. 1, that first relies on a continuous sign language recognition model to produce sign glosses and then passes the produced glosses into a neural machine translation (NMT) system to generate text translations. Glosses are used by sign language linguistics and annotators as a written form or word-to-word translations of sign languages. Although they share parts of the vocabulary of the

Xuan Zhang
Johns Hopkins University, USA, e-mail: xuanzhang@jhu.edu

Kevin Duh
Johns Hopkins University, USA e-mail: kevinduh@cs.jhu.edu

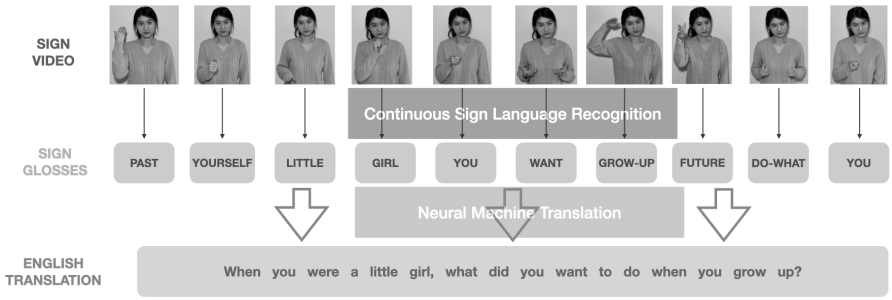

Fig. 1 A cascaded sign language translation system. A sign language video is first converted to a sequence of glosses with a continuous sign language recognition model. A neural machine translation model then maps glosses to spoken language text translations.

spoken language, it reflects the linguistic characteristics of sign languages which are syntactically different from their spoken language counterparts [1]. Therefore, glosses should be treated as another language, and mapping glosses to the text of spoken languages is a valid NMT task. In this chapter, we focus on studying methods that could potentially improve the performance of gloss-to-text translation, which is the NMT component of cascaded systems.

Sign language gloss translation is a challenging problem due to the scarcity of annotated parallel data. NMT systems are often extremely data-hungry and usually require millions of training examples to achieve the state-of-the-art translation performance, while the most widely used dataset for SLMT, RWTH-PHOENIX-Weather 2014 [2], contains only 7,096 video-gloss-text triplets in the training set. Our goal is to examine the commonly-used techniques in low-resource machine translation for this task (Sect. 2), and analyze how the performance of gloss translation contributes to the performance of sign language translation overall (Sect. 3).

2 Gloss Translation as a Low-Resource Translation Task

In this section, we treat gloss-to-text translation as a low-resource machine translation task and explore the effectiveness of various commonly-used techniques on two real sign language datasets.

2.1 Low-Resource Machine Translation

Recent state-of-the-art NMT models such as the Transformer [3] follow an encoder-decoder architecture. The conditional probability of generating the target sentence y given the source sentence x is decomposed as:

$$p(y \mid x) = \prod_{j=1}^{J} p(y_j \mid y_{<j}, x, \theta), \tag{1}$$

where θ represents model parameters, J is the length of the target sentence, y_j is the j-th target word, and $y_{<j}$ is the prefix of words before y_j. The encoder of an NMT model transforms x into a sequence of hidden states, the decoder then generates y_j iteratively based on the hidden states and the history decoding states to form the target sentence y. In the context of sign language gloss translation, x is the gloss and y is the text translation. For example in Fig 1, x is the American Sign Language (ASL) gloss sequence "PAST YOURSELF LITTLE GIRL YOU WANT GROW-UP FUTURE DO-WHAT YOU" and y is the English text sequence "When you were a little girl, what did you want to do when you grow up?"

The training objective of NMT model is minimizing the cross-entropy loss, which is a measure of the difference between the reference token and predicted target token, which is often represented as a probability distribution.

Low-resource machine translation is the task of machine translation in the settings when there is a limited amount of training data, which is in the order of 10,000 or less. This topic has received a lot of attentions in the machine translation research community. There are several methods that have been adopted to improve the performance of machine translation in low-resource scenarios [4].

1. Data collection. One way to deal with the challenges of data scarcity is collecting more data, either by web-crawling for parallel data or annotating monolingual data. The web-crawling result is often noisy, whose quality is not guaranteed, and thus needs careful cleaning and preprocessing. In the case of gloss translation, web-crawling is likely infeasible due to the lack of gloss transcriptions on the web. Similarly challenging, annotating in many cases requires the involvement of human experts, which makes it expensive and time-consuming.
2. Monolingual data exploitation. Compared to collecting reliable parallel data, a much cheaper way to increase the amount of data is to generate synthetic data. Monolingual data is often abundant; for gloss translation, this is especially true on the target side. Back-translation is one effective way to leverage monolingual data. Another approach is integrating external language models into NMT models. Besides, monolingual data can also be utilized using transfer learning, where pretrained embeddings or pretrained language models are used to initialize part of the NMT systems.
3. Multilingual data exploitation. A multilingual model is trained to be a universal model that is capable of translating between any two languages. Model parameters are shared across multiple language pairs, and low-resource ones may benefit from knowledge learned from others. There are off-the-shelf pretrained multilingual machine translation models which can be used as the initialization for further training on low-resource language pairs. Alternatively, a multilingual model can also be trained from scratch with the low-resource language pairs included in the training set.
4. Model choice. Improved architectures and training methods are effective in tackling low-resource scenarios. These include hyperparameter search, meta-learning

for multilingual learning, incorporating a latent variable to capture linguistically-motivated inductive bias, or using alternative training objectives instead of cross-entropy loss to alleviate the exposure bias, etc.

2.2 Datasets and Experiment Setup

As the sizes of sign language datasets that have human-annotated gloss transcriptions are all in the order of 10,000, the gloss and text translation of sign language can be considered as a low-resource language pair. Thus, we examine the most commonly-used methods in Sect. 2.1 and apply them to gloss translation.

In the following, we explore the effectiveness of various low-resource machine translation approaches on gloss to text translation and discuss the potentials and pitfalls of each of the method. We first introduce the datasets used in the experiments involved in the rest of this section (Sect. 2.2). We start our exploration of different methods with hyperparameter search (Sect. 2.3). We then move to fine-tune the pretrained multilingual models (Sect. 2.4), which leads to a significant boost in performance. Based on the pretrained models, we study how data augmentation would further improve the results. Specifically, we look at generating synthetic data by human-crafted rules (Sect. 2.5) and back-translation (Sect. 2.6). While additional synthetic data would introduce noisiness into the training data, curriculum learning (Sect. 2.7) arranges the presenting order of the samples such that the model learns more effectively and reaches higher levels of performance. These methods can be categorized using the taxonomy proposed in [4] as in Fig. 2.

The experiments are conducted on two sign language datasets, RWTH-PHOENIX-Weather 2014T and CSL Daily.

1. **RWTH-PHOENIX-Weather 2014T.** The PHOENIX14T dataset is collected from the weather forecast airings of the German public TV station PHOENIX. It is by far the most widely used benchmark dataset for SLMT. It contains RGB German sign language (DSL) videos performed by nine signers, gloss annotations and German translations. The data split for train/dev/test set is 7,096/519/642 samples. The vocabulary size of the training set for glosses and German are 1,066

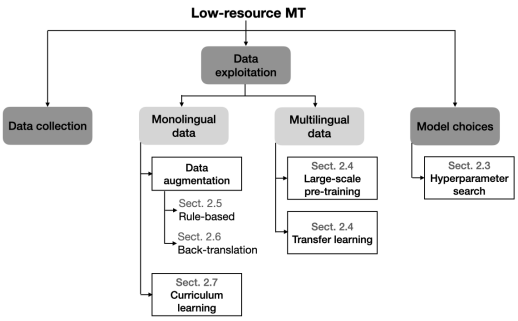

Fig. 2 Methods for low-resource machine translation explored in Sect. 2 categorized under the taxonomy introduced in [4].

and 2,887 respectively. On average, there are 6.6 words in glosses and 13.2 words in German translations.

2. **CSL Daily.** CSL Daily [5] is a Chinese sign language (CSL) dataset. It covers a wide range of topics encountered regularly the Deaf community in their daily life, including family life, medical care, school life, bank service, shopping, social contact and so on. There are 10 signers involved in the recording. CSL Daily also provides both the gloss and Chinese translation annotations. The size of train, dev and test set is 18,401, 1,077 and 1,176 respectively. The vocabulary size of the training set is 2,000 for glosses and 2,343 for Chinese translations. The average number of Chinese characters in glosses is 11.1 and 15.8 in Chinese translations.

2.3 Hyperparameter Search

Hyperparameter selection is crucial to build a good NMT system. It is especially the case for low-resource scenarios when the default hyperparameter settings are very likely to be ineffective. As reported in [6] and [7], the NMT systems developed for low-resource translation tasks disagree a lot with those trained on high-resource corpora on the optimal hyperparameter choices. Furthermore, datasets in different domains and language pairs all differ in their hyperparameter preference. It is also reported that adjusting hyperparameters can improve the BLEU score by 20 in some datasets.

2.3.1 Important Hyperparameters

Here we focus on 4 hyperparameters of Transformer models: the number of BPE merge operations, the number of layers, embedding dimensions and initial learning rate. These hyperparameters are recognized as important hyperparameters by [7], where the importance is computed as the variation in BLEU when changing a specific hyperparameter with values of all the other hyperparameters fixed [8].

BPE is a word segmentation approach that combines frequent sequence of characters so that out-of-vocabulary words are handled. It is expected to improve the translation of rare words and has been a standard preprocessing practice in NMT. According to [9], although 32k and 90k are popular choices in most machine translation literature, they found that the BPE of the best Transformer-based architectures in low-resource setting is somewhere between 0-2k. We thus try 1k and 2k in our experiments.

Architecture design hyperparameters like the number of layers in encoder and decoder and embedding size are important. A big and complex model is more susceptible to overfitting. On the other hand, if the model is too small and simple, it might struggle to capture the meaningful patterns of data and result in underfitting. Our hyperparameter search space includes 1, 2, 4 layers and embedding size of 256 and 512.

The learning rate is another important hyperparameter that scales the gradient in gradient descent training. A small initial learning rate may prolong the training process, whereas a large one may get the model stuck in a sub-optimal solution. It is recommended to start training with a low number [10]. We adjust it among 0.00005, 0.0005 and 0.0002.

2.3.2 Results

We tune hyperparameters for Transformers trained on PHOENIX14T which are summed up to 36 systems in total. The BLEU scores obtained on the search space are illustrated in Fig. 3. It shows a nice bell curve, where the majority of the Transformer models lie in the range of BLEU scores between 22.40 and 23.60, with fewer models on either side further away from the center. Table 1 shows the best, worst and a random system obtained through hyperparameter search. Compared to other models, the best model has the smallest BPE operations (1k between 1k and 2k), the largest number of layers (4 among 1, 2 and 4), the largest embedding size (512 between 256 and 512) and the smallest initial learning rate (0.00005 among 0.00005, 0.0005 and 0,0002), which leads to a 2.65 BLEU score improvement over the worst model. The random model in Table 1 simulates the situation when randomly picking a hyperparameter configuration without adjusting it. According to the distribution of the models as illustrated in Fig. 3, when randomly sampling a model from the search space of the hyperparameter configurations, the performance of the model will be most probable within the range of 22.40 and 23.60 BLEU score. This is around 1 BLEU score lower than that of the best configuration. In other words, in most of the cases, it is assured to get a superior model with hyperparameter search.

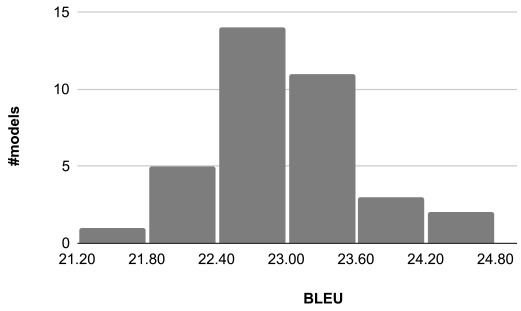

Fig. 3 The performance distribution of Transformers trained on PHOENIX14T gloss-text data with 36 different hyperparameter configurations. It is roughly a bell curve with more than half of the models centered around 22.40 to 23.60 BLEU score, and fewer models scattered further from the center.

2.3.3 Discussions

Hyperparameter search can significantly improve the performance of the NMT model. The optimal hyperparameter configurations may be different from dataset

Table 1 Performance of selected Transformers on PHOENIX14T gloss-text data. **Best** and **worst** are the best and worst systems obtained from hyperparameter search. **Random** randomly picks a hyperparameter configuration from the search space whose size is 36. The **best** and **worst** model differ in 3 out of the 4 hyperparameters that have been tuned and result in a 2.65 BLEU difference.

	bpe	#layer	#embed	init_lr	**BLEU**
best	1k	4	512	5e-5	**24.38**
worst	2k	1	512	5e-4	**21.73**
random	1k	2	256	2e-4	**23.49**

to dataset or from task to task. There is no straightforward way to determine the best settings without actually running the experiments. Tuning hyperparameters is often computationally expensive and time-consuming, but it is not the case for low-resource NMT scenarios, where the training can be finished within hours on a single GPU.

We have only covered 4 hyperparameters so far, but in practice, it is nontrivial to try a larger search space in terms of both the types and values of hyperparameters.

There are a number of other hyperparameters that can be tuned when training a Transformer model. The number of hidden nodes in feed-forward layers and the number of attention heads can be classified as the model architecture as the number of layers and the embedding size or model size, as they determine the number of model parameters or the complexity of the model. The width of the feed-forward layer affects the Transformer's ability to learn features from the input data and should not be excessively large or small to avoid overfitting or underfitting. The number of attention heads allow the model attend to different aspects of the features simultaneously. Sometimes, a smaller number of attention heads are sufficient to achieve good performance, and using a larger number of attention heads might not provide any additional benefit. Training configurations, for example, batch size, label smoothing, dropout, optimizer can also matter. Batch size is the number of samples to be processed simultaneously by the model. A larger batch size can lead to faster training and more stable gradient updates but may also make the model prone to overfitting. Label smoothing and dropout are both regularization techniques preventing overfitting. In label smoothing, the model is predicting soft labels. Dropout works by setting a hidden node to zero with a dropout probability. An optimizer is an algorithm that is designed to adjust the model parameters during training based on the gradients of the loss function. Common optimizers include stochastic gradient descent(SGD), Adam, Adagrad, Adadelta and RMSprop, which differ in the sensitivity to the learning rate and robustness to the noise in the gradients. Last but not the least, there are two other hyperparameters that are important but frequently overlooked by the practitioners: the checkpoint interval and random seed. The checkpoint interval is the interval to save the checkpoint and evaluate the model on validation set. When training with a smaller dataset, it is always helpful to have a smaller checkpoint interval but it might also prolong the training. The choice of random seed influences weight initialization and training data order, and [11] finds that some weight initializations and data orders perform better than others.

With regard to the hyperparameter values to try, there is a trade-off in deciding both the range and granularity of the values. First, we might expand on a wider range of values (e.g., change #hidden = $\{1024, 2048\}$ to $\{512, 1024, 2048, 4096\}$). Second, we might expand on a more fine-grained range of values (e.g., change #hidden = $\{1024, 2048\}$ to $\{1024, 1536, 2048\}$). Although wider range and finer granularity are desirable, each additional value causes an exponential increase in the number of models because of the cross-product of all values. The decisions need to be made based on the available computational resources.

2.4 Pretrained Multilingual Models

Pretraining and fine-tuning has become a widely used approach in natural language processing (NLP). In this section, we adopt this approach for the gloss-to-text translation task. We will use the pretrained Transformer-based multilingual model mBART and fine-tune it on our sign language datasets.

2.4.1 mBART

The large-scale self-supervised pretraining on Transformer-based models has led a great progress in NLP. In particular, BERT [12] advances the state-of-the-art results in a variety of NLP benchmarks, such as GLUE [13] and SuperGLUE [14]. There are also other large pretrained language models developed after BERT, for example, GPT [15], XLNet [16], RoBERTa [17] and T5 [18], which further push forward the baselines. Though not simply the variants, they all share similarity with BERT. Typically, these models are first pretrained on large corpora, then fine-tuned on smaller task-specific downstream datasets by using the model parameters as initialization, and adding extra task-specific layers on top of the architecture. Pretraining can be beneficial because it captures a wide range of statistical patterns and regularities in the language and learns useful representation of the language. It makes the fine-tuning more efficient compared to training from scratch with random initialization of model parameters.

BERT, GPT, XLNet, RoBERTa and T5 are all trained on copora in a single language, such as English, and focus on pretraining parts of the Transformer model, either on the encoder (BERT, XLNet, RoBERTa) or the decoder (GPT, T5). MBART [19] instead trains the entire Transformer model on large-scale monolingual corpora across many languages. Compared to BERT, which is more suitable for language understanding tasks, mBART is designed for machine translation tasks. It sets new state-of-the-arts on multiple machine translation benchmarks.

MBART follows the same pretraining scheme as in BART [20], while BART focuses only on English. The BART is trained as a denoising autoencoder: text is first corrupted with an arbitrary noising transformation, such as masking or deleting random tokens, permuting sentences in documents, rotating the documents, and the

goal of the pretraining is to recover the original text. MBART applies BART to many languages by adding a language id symbol to each of the instances.

2.4.2 Results

We use the pretrained mBART and fine-tune it on the PHOENIX14T dataset. Since mBART has not been trained on gloss, we treat gloss as German in the input. Table 2 summarizes the results. It shows that by initializing the model parameters with pretrained mBART, the performance improves significantly upon the best model we obtained through hyperparameter search on the models trained from scratch.[1] This improvement comes from the fact that the encoder and decoder are pretrained simultaneously in mBART and it is pretrained on large corpora in many languages, which results in more fluent translations.

Table 2 The performance of translation models on PHOENIX14T with and without the initialization of mBART. BLEU scores on the test set are reported. **G2T w/o mBART** is a gloss-to-text translation model trained from scratch, while **G2T w/ mBART** is initialized with pretrained mBART and is fine-tuned on PHOENIX14T. The results on **S2G2T** and **S2T** models are from [21], with the former as a two-stage sign language translation system and the latter as an end-to-end system. The text translation component of these two models are either initialized with parameters from a Transformer trained on PHOENIX14T (**w/o mBART**) or a pretrained mBART (**w/ mBART**). Using mBART significantly improves the performance.

	G2T	**S2G2T**	**S2T**
w/o mBART	24.38	20.17	23.28
w/ mBART	26.70	24.60	28.39

2.5 Rule-Based Synthetic Data Generation

Although parallel gloss-text data are scarce and large gloss corpora created from transcribing the sign languages are also not available, there are an abundant amount of monolingual data in spoken languages. Using human-crafted rules, we can generate pseudo-glosses and pseudo-parallel gloss-text samples from spoken languages as additional training data for gloss-to-text translation. In this section, we test the effectiveness of this approach on both PHOENIX14T and CSL Daily.

[1] The previous hyperparameter search does not include the hyperparameter configuration for mBART, which contains 12 layers of encoder and 12 layers of decoder, and would easily overfit if trained on a small dataset exclusively.

2.5.1 Rules

Glosses share parts of the vocabulary of their spoken language counterparts, and are mainly different from spoken languages on the following aspects:

1. A lack of word inflection. In glosses, words do not exhibit inflectional morphological processes to reflect the tense, aspect, person, gender and case.
2. An omission of selected words. In glosses, the articles (e.g. “a”, “an”, “the”) are always omitted. Prepositions might be omitted when they indicate the time (e.g. “The book will be published *in* next year”), the direction or movement (e.g. “She went *to* the museum”), association (e.g. “They left because *of* the noises”, “She is the CEO *of* the company”). Sometimes, the subject and/or object might be omitted, if they can be inferred from the context. All forms of verb “be”, such as “is”, “are” and auxiliary verbs are also omitted (e.g. “She *has* finished her homework”, “*Do* they want to eat”).
3. Word order. Glosses employ the strategy of topicalization, where a word or a phrase is placed at the beginning of a sentence to emphasize that it is the focus of the sentence. [2] Glosses also have the feature of subject pronoun copy, in which a pronoun is repeated at the end of the sentence to indicate the subject. [3] Besides, in glosses, the question words (e.g. “who”, “what”, “where”, “why”) are often used as postpositions to indicate the type of information being sought in questions. There are other syntactic features that are unique in sign languages glosses, which can vary significantly from one language to another.

For example, in Fig. 1, the words in the sign gloss are all lemmas. The article “a”, the preposition “to”, “were” and the auxiliary verb “did” are omitted. The subject pronoun (“YOU”) is repeated at the end of the gloss. And the question word (“DO-WHAT”) is also postpositioned.

Accordingly, [1] propose the following three heuristic rules to generate pseudo-glosses from spoken languages: (1) Lemmatization of spoken words; (2) POS-dependent and random word deletion; (3) Random word permutation. We follow [1] for our experiments. More specifically, we generate the pseudo-gloss with slightly different processes for DSL and CSL respectively as follows.

Generate DSL from German

For each German sentence S,

```
1. Lemmatize all the words in S and keep words only if their
Part-of-speech (POS) tags are in the set of {VERB, PRON,
PROPN, ADJ, ADV, NOUN, NUM}.

2. If the length of S is greater than 5, remove words by
```

[2] Topicalization is an important feature for many spoken languages such as Japanese. Here, we mainly consider the difference between American Sign Language and English.

[3] Note that subject pronoun copy is not used in all the sign languages.

```
the probability of 0.1.

3. Permute the words with the probability of 0.5 with
permutation distance smaller or equal to 4.
```

To generate CSL glosses, we adopt a more language-specific strategy. We manually build a dictionary that maps words in spoken Chinese to vocabulary in CSL glosses. The dictionary has 250 entries, the mappings are learned from the CSL Daily dataset. Note that the dictionary might not be a thorough list of all the possible mappings. Fig. 4 shows some entries in the dictionary.

Fig. 4 A dictionary that maps words in spoken Chinese to vocabulary in CSL glosses. The vocabulary of CSL glosses is a subset of the vocabulary of spoken Chinese. Some particles and articles are omitted in glosses. In addition, it is often the case that a set of synonyms in spoken Chinese is represented by a single gloss.

Chinese	CSL	English Translation
了/ 吗 / 的	-	*Particles*
只 / 支 / 件	-	*Articles*
家	房子	Home
漂亮	美丽	Beautiful
开车	驾驶	To drive
上班	工作	To work
父母	爸爸妈妈	Parents
对手	敌人	Enemy

Generate CSL from Chinese

For each Chinese sentence *S*,

```
1. Replace words in S that appear in the Chinese-CSL dictionary.

2. Remove punctuations.

3. For each of the word that has POS tag ‘‘VERB", move the word
after the object of the verb with the probability of 0.5.

4. For each of the word that has POS tag ‘‘NUM", convert
Chinese numericals to Arabic numericals.

5. Permute S such that the numbers, verbs and negation words
are at the end of the sentence.
```

2.5.2 Data Selection

It is not trivial to carefully choose the source of the monolingual data. Web-crawled data are noisy and have a lot of randomness. For example, web pages often contain HTML tags and formatting, non-textual data such as images and videos and unrelated information such as emails, phone numbers and hyperlinks. Web pages may also contain low-quality content and a mix of different language variations and slang terms. On the contrary, publicly available datasets that have been cleaned and preprocessed by experts are relatively in higher quality. However, those datasets might also contain sentences that are irrelevant or even harmful, making the training less efficient.

Typically, we want the monolingual data to be within the same domain as our parallel data. The reason is that out-of-domain corpora have different word distributions and expression styles from the in-domain data, which would cause unstable training and dataset shift, where there is a mismatch between the distribution of training and test set.

We can still utilize the out-of-domain data by data selection, which involves selecting a subset from the out-of-domain data that is most similar to the in-domain data. There are many existing data selection methods. We adopt the **Moore-Lewis** approach proposed in [22]. The main idea is to score the out-of-domain data N using language models trained from the in-domain data I and N separately, and select top n examples from N by a cut-off threshold on the resulting scores. To be specific, each sentence s in N is assigned a cross-entropy difference score,

$$H_I(s) - H_N(s), \tag{2}$$

where $H_I(s)$ is the per-word cross-entropy of s according to a language model trained on I, and $H_N(s)$ is the per-word cross-entropy of s according to a language model trained on N. A lower score indicates s is more like a sentence in I then in the domain of N.

2.5.3 Results

For DSL-to-German translation, we use the News crawl from the shared translation task of WMT21 [23] as the German monolingual data. It is a collection of large corpora of crawled news, collected since 2007. It contains more than a hundred million of German sentences in total. For CSL-to-Chinese translation, we combine two datasets: DailyDialog [24] and LCCC-large [25] . The DailyDialog is a multi-turn daily dialogue dataset which contains 96,784 sentences. The dialogues in the dataset are written by human and cover various topics about daily life. The LCCC-large is a large-scale cleaned Chinese conversation dataset, which contains 12 million dialogues.

We run Moore-Lewis selection and rank the sentences with Eq. 2. We keep the top 50k sentences and discard the rest. We then generate the pseudo-gloss to make

Table 3 Gloss-to-text NMT models on PHOENIX14T. BLEU scores on test set are reported. All models are initialized with pretrained mBART. **Baseline** performs continued-training on PHOENIX14T. *Rule-based* models are trained on the combination of PHOENIX14T and selected synthetic data generated by human-crafted rules, while *Back Translation* models use the concatenation of PHOENIX14T and selected parallel data generated by back translation. *Fine-tuning* models are fine-tuned on PHOENIX14T. *Curriculum Learning* models apply curriculum learning when training on the enlarged dataset. Naively training on the synthetic data as in *Rule-based* and *Back Translation* hurts the performance, while *fine-tuning* and *curriculum learning* make the model utilize the out-of-domain data more effectively and always help enhance the performance.

Model	**BLEU** (*imp. over baseline*)
Baseline	26.70
Rule-based	26.52 (*-0.18*)
Rule-based + Fine-tuning	27.17 (*+0.47*)
Rule-based + Curriculum Learning	27.40 (*+0.70*)
Rule-based + Curriculum Learning + Fine-tuning	27.66 (*+0.96*)
Back Translation	25.77 (*-0.93*)
Back Translation + Fine-tuning	28.34 (*+1.64*)
Back Translation + Curriculum Learning	28.09 (*+1.39*)
Back Translation + Curriculum Learning + Fine-tuning	**28.60 (*+1.90*)**

additional parallel data. Fig. 5 shows samples with the highest Moore-Lewis scores in Chinese.

CSL Daily		Top 8 Selected Sentences	
Text	**Ref Gloss**	**Text**	**Pseudo-gloss**
他是谁？(Who is he?)	他 谁	很高兴认识你。(Nice to meet you.)	高兴 你 认识
你是老师吗？(Are you a teacher?)	你 老师 是	你是个吃货！(You are a foodie.)	你 吃货
他是我的手语老师。(He is my sign language teacher.)	他 我 手语 老师	你的猫多大了？(How old is your cat?)	你 猫 年龄 多少
我给孩子取名字。(I am giving my child a name.)	我 名字 孩子	发展才是硬道理。(Development is the hard truth.)	发展 才 硬 道理
小猫在哪里？(Where is the kitten?)	小 猫 在 哪	你去做什么？(What are you going to do?)	你 做 去 什么
你家有几个人？(How many people are there in your family?)	你 房子 人 多少	你想吃什么？(What do you want to eat?)	你 想 吃 什么
你女儿几岁？(How old is your daughter?)	你 女儿 年龄 多少	你叫什么名字？(What's your name?)	你 名字 什么
我们什么时候去看电影？ (When are we going to watching a movie?)	我们 看 电影 什么 时间	最近有什么好看的电影？ (What are some good movies to watch recently?)	近 好看 电影 什么

Fig. 5 Examples of gloss-text pairs in CSL Daily and sentences selected from DailyDialog and LCCC-large. The 8 selected sentences rank highest according to the score defined in Eq. 2. They are considered to be most similar to CSL Daily. Pseudo-glosses are generated based on human-crafted rules.

We initialize our NMT models with pretrained mBART and train them on the concatenation of in-domain (PHOENIX14T or CSL Daily) and selected out-of-domain data. Fine-tuning is then optionally performed on the in-domain bitext only, leading to the following 2 systems:

Rule-based Continue training the pretrained mBART on the concatenation of in-domain and selected out-of-domain data.

Table 4 Gloss-to-text NMT models on CSL Daily. BLEU scores on test set are reported. Model setups are similar to Tab. 3 except that back translation is conducted on either *NMT* or *SMT*.

Model	**BLEU** (*imp. over baseline*)
Baseline	29.68
Rule-based	28.25 (*-1.43*)
Rule-based + Fine-tuning	30.00 (*+0.32*)
Rule-based + Curriculum Learning	29.91 (*+0.23*)
Rule-based + Curriculum Learning + Fine-tuning	29.83 (*+0.15*)
Back Translation w/ NMT	28.58 (*-1.10*)
Back Translation w/ NMT + Fine-tuning	29.04 (*-0.64*)
Back Translation w/ SMT	27.96 (*-2.28*)
Back Translation w/ SMT + Fine-tuning	29.50 (*-0.18*)
Back Translation w/ NMT + Curriculum Learning	29.42 (*-0.26*)
Back Translation w/ NMT + Curriculum Learning + Fine-tuning	29.89 (*+0.21*)
Back Translation w/ SMT + Curriculum Learning	**30.42 (*+0.74*)**
Back Translation w/ SMT + Curriculum Learning + Fine-tuning	30.18 (*+0.58*)

Rule-based + Fine-tuning Fine-tune the **Rule-based** model on in-domain data.

Tab. 3 and Tab. 4 compare the two rule-based systems on PHOENIX14T and CSL Daily respectively, where the **baseline** model is the model obtained in Sect. 2.4, which is a pretrained mBART fine-tuned on the in-domain data. Training on the expanded dataset without fine-tuning hinders the performance. The might be due to the low-quality of the selected out-of-domain data or the generated pseudo-gloss, such that the trained model is not well-suited to the task in the target domain. Fine-tuning on the other hand adapts the model to the target domain and let the model take advantage of enriched training data.

2.6 Back-translation

Another way to create synthetic parallel data using monolingual text is back translation, in which the target language is translated back into the source language such that the bitext is generated. The workflow of back translation is illustrated in Fig. 6.

2.6.1 Results

We obtain the additional parallel data as follows. (1) Select the out-of-domain monolingual text following the same procedure as Sect. 2.5.2. (2) Train a text-to-gloss model from scratch on in-domain data (PHOENIX or CSL Daily). (3) Translate monolingual text to gloss. For step (2), the reliability of the model is important since it affects the quality of the generated data. NMT models such as Transformer is effective with sufficient data. On the contrary, statistical machine translation (SMT) models tend to be more efficient when workin with low-resource settings. This is be-

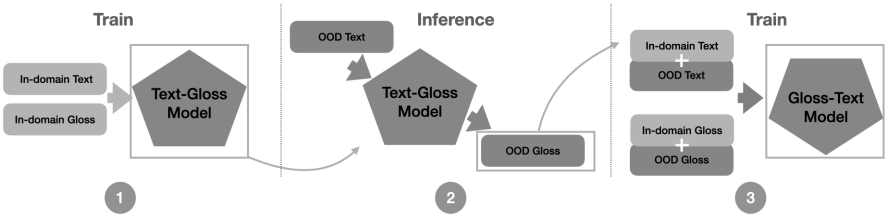

Fig. 6 The workflow of back translation. First, a text-to-gloss model is trained on in-domain data. Second, the text-to-gloss model translates out-of-domain (OOD) text into glosses. Finally, train a gloss-to-text model on the concatenation of in-domain and translated OOD data.

cause SMT is based on the use of statistical models which can be built using a small amount of data. Therefore, we try both NMT and SMT for text-to-gloss models.

The experiment setup is simialr to Sect. 2.5.3. We initialize our NMT models with pretrained mBART and train them on the concatenation of in-domain (PHOENIX14T or CSL Daily) and synthetic parallel data generated by back translation using NMT or SMT. Fine-tuning is optionally performed on the in-domain bitext, leading to the following 4 systems:

Back Translation Same as **Back Translation w/ NMT**. Continue training the pretrained mBART on the concatenation of in-domain and selected synthetic data generated by back translation with NMT.

Back Translation + Fine-tuning Same as **Back Translation w/ NMT+ Fine-tuning**Fine-tune the **Back Translation** model on in-domain data.

Back Translation w/ SMT Continue training the pretrained mBART on the concatenation of in-domain and selected synthetic data generated by back translation with SMT.

Back Translation w/ SMT + Fine-tuning Fine-tune the **Back Translation w/ SMT** model on in-domain data.

Results on PHOENIX14T and CSL Daily are shown in Tab. 3 and Tab. 4 respectively. On PHOENIX14T, back translation without fine-tuning degrades the performance. Adding the step of fine-tuning improves the performance. This is consistent with the observations for rule-based systems, while back translation with fine-tuning outperforms rule-based system with fine-tuning. On CSL Daily, back translation systems all hurt the performance even with fine-tuning.

2.6.2 Analyses

Tab. 4 is discouraging to justify the effectiveness of back translation. Is it because the selected monolingual data is substantially different from the in-domain data

or the text-to-gloss model can not output good translations? Or is it because back translation do not work at all?

To answer those questions, we evaluate back translation on a much simpler situation, where domain mismatch is not a concern. To be specific, we split PHOENIX14T evenly into two subsets, part 1 and part 2, which are treated as in-domain and out-of-domain data respectively. We discard the gloss of part 2 and instead generate pseudo-gloss using a text-to-gloss model trained on part 1. We evaluate the performance of a gloss-to-text model trained from scratch on the combination of part 1 and the synthetic part 2. Results are shown in Tab. 5. It can be seen that although the text-to-gloss model is bad (*T2G on part1*), back translation can still enhance the model performance (*G2T on part1+synthetic part2* vs. *G2T on part1*). We can conclude that back translation is helpful with ghigh-quality monolingual data.

Table 5 Back translation using PHOENIX14T data only.The training set of PHOENIX14 is evenly split into two subsets, part1 and part2. The gloss of part2 is discarded and then recovered by back translation. With the synthetic part2, the gloss-to-text model outperforms the one trained only on part1, which concludes that back translation is effective when the monolingual data is within the same domain of parallel data.

NMT systems	**BLEU**
G2T on part1	19.13
T2G on part1	9.96
G2T on part1+synthetic part2	21.57

2.7 Curriculum Learning

The order in which the data is presented to the NMT model affects the learning process. If the composition of data is relatively complex and the data is presented in an random manner, it may be harder for the model to learn certain patterns or representations. This happens when we incorporate the synthetic data into training set in Sect. 2.5 and Sect. 2.6. Alternatively, if data is presented in an organized manner, it may be easier for the model to learn.

Curriculum learning is a method that gradually increases the complexity of samples used during the training process. It has been successfully employed in many tasks in NLP. An implementation of curriculum learning should resolve two questions. (1) How to rank the training examples, or in other words, how to define the complexity of a sample. (2) How to modify the sampling procedure based on the ranking. We adopt the curriculum learning training strategy proposed by [26] and [27], which is proven effective on a domain adaptation task for NMT with a similar low-resource setting as ours.

2.7.1 Training Strategy

In this section, we briefly introduce the curriculum learning training strategy used in [27]. [27] defines the complexity of a sample based on its similarity to the in-domain data, which can be quantified by Eq. 2. They employ a *probabilistic* curriculum, in contrast to the *deterministic* curriculum, which trains on a fixed order of samples based on their scores. The deterministic curriculum may always perform well because neural models benefit from randomization in the minibatches and multiple epochs. The probabilistic curriculum works by dividing the training procedure into distinct phases. Each phase creates a random sample from the entire pool of data, but earlier phases sample the "easier" or "more similar to in-domain" sentence with higher probability. Since each phase can be viewed as creating a new training dataset, all the well-tested tricks of the trade for neural network optimization can be employed. The training strategy (Fig. 7) is summarized as follows:

- Sentences are first ranked by similarity scores and then distributed evenly into shards, such that each shard contains samples with similar similarity criteria values.
- The training process is segmented into consecutive phases, where only a subset of shards are available for training.
- During the first phase, only the easiest shard is presented. When moving to the next phase, the training set will be increased by adding the second easiest shard into it, and so on. Easy shards are those that are more similar to the in-domain data, as quantified by Moore-Lewis (Eq. 2).
- The presentation order of samples is not deterministic. (1) Shards within one curriculum phase are shuffled, so they are not necessarily visited by the order of similarity level during this phase. (2) Samples within one shard are bucketed by length and batches are drawn randomly from buckets.

2.7.2 Results

We use the same experiments setups as in Sect. 2.5 and Sect. 2.6 except that when training on the concatenation of in-domain and synthetic parallel data, the curriculum learning training strategy is applied, resulting in the *Curriculum Learning* models in Tab. 3 and Tab. 4:

- Curriculum learning significantly improves the performance. It is used in the best models for both PHOENIX14T and CSL Daily: Tab. 3 *Back Translation + Curriculum Learning + Fine-tuning* and Tab. 4 *Back Translation w/SMT + Curriculum Learning*.
- Curriculum learning is able to make better use of data with complex compositions. In Tab. 3, the *Back Translation* model performs worse than the *Rule-based* model

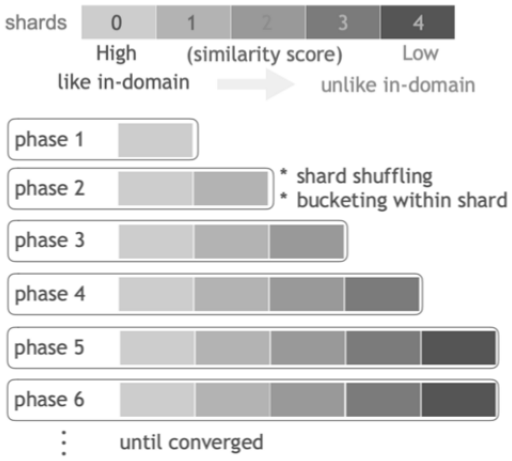

Fig. 7 The curriculum training strategy proposed in [26]. Sentences are ranked by similarity scores and distributed evenly into shards. The training process is segmented into consecutive phases, in each of which the model is trained on a subset of shards. The training set is increased gradually by adding shards with samples of lower similarity scores into it. Note that the presentation order of samples is not deterministic. (1) Shards within one curriculum phase are randomly shuffled. (2) Samples within one shard are bucketed by length and batches are drawn randomly from buckets.

(25.77 vs. 26.52). However, with curriculum learning, this relationship has shifted. The *Back Translation + Curriculum Learning* outperforms the *Rule-based + Curriculum Learning* model (28.09 vs. 27.40). Another example is the *Back Translation w/ NMT* and *Back Translation w/ SMT* in Tab. 4.

- Fine-tuning after curriculum learning does not always lead to a better model, for example, in Tab. 4, *Rule-based + Fine-tuning* has a higher BLEU score than *Rule-besed + Curriculum Learning + Fine-tuning* (29.91 vs. 29.83).

3 Gloss Translation as a Part of Sign Language Translation

In general, there are two types of architecture for a sign language translation system (Fig. 8). (1) A cascaded system (the focus of this work) as shown in Fig. 1 that trains a sign language recognition model and a gloss-to-text translation model separately. During inference, signs are first mapped to gloss by the recognition model, and the gloss outputs are then translated to text in spoken language with the translation model. (2) An end-to-end system, that is usually constructed with a visual encoder that learns visual representations from sign videos and an encoder-decoder structure that decodes the representations into text translation, in which the latter can be initialized with the parameters from training on reference gloss and text. Gloss may be added as another supervision signal using CTC loss, which is short for Connectionist Temporal Classification loss, helping the model learn to map input sequences of variable length to output sequences of fixed length.

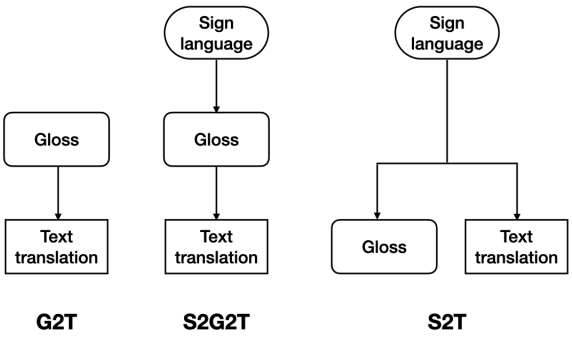

Fig. 8 Illustration of architectures of sign language translation models. G2T translates gloss to text. S2G2T is the cascaded system that first recognizes sign to gloss and then translates gloss to text. S2T is the end-to-end system that translates sign to gloss and text simultaneously.

Tab. 2 shows that the improvement on gloss-to-text (G2T) can transfer to both the cascaded system (S2G2T) and end-to-end system (S2T) and can even be amplified. With an 2.32 BLEU score improvement on G2T, the BLEU increases by 4.43 on S2G2T and 5.11 on S2T.

It seems that the performance of gloss translation contributes a lot to the overall performance of a sign translation system. But how about errors? Does G2T also contribute a lot to the errors made by S2G2T and S2T? Or Does it simply pass along the errors from the sign recognition component (S2G)? Is it even able to fix some errors made by S2G? In next subsection, we will try answering these questions by error analyses.

We conduct error analyses based on statistics and case studies on the outputs from S2G, G2T, S2G2T and S2T trained on CSL Daily, in which S2G2T is formed by S2G and G2T, while S2G2T is initialized by the pretrained checkpoints of S2G and G2T. The checkpoints used are from [21].

3.1 Statistics

We collect statistics about the translation accuracy in terms of token and POS.

Token Recall

The token recall is calculated as the number of correct predictions of a certain token divided by the number of total appearance of that token in the reference gloss or text translation. It can be considered as a more in-depth performance metric than BLEU. Fig. 1 illustrates the recall distribution indexed by token appearance frequency. S2G has a higher-than-60% recall for most of the tokens. G2T instead has a more scattered distribution, which is inherited by S2G2T and S2T. This indicates that building a successful SLTM model highly depends on G2T.

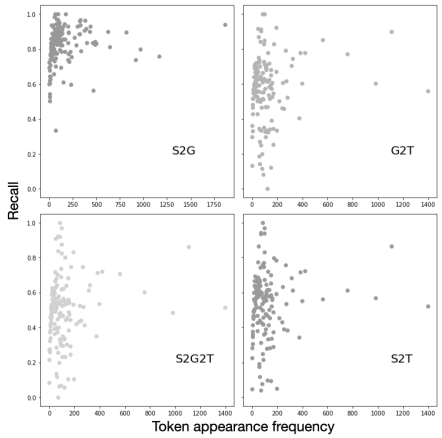

Fig. 9 Recall for tokens on S2G, G2T, S2G2T and S2T systems. Recall is calculated as the number of correct predictions of a certain token divided by the number of total number of the appearance of that token in the reference. S2G has a higher token-level recall compared to the others, and the token distribution on S2G2T and S2T are more similar to G2T.

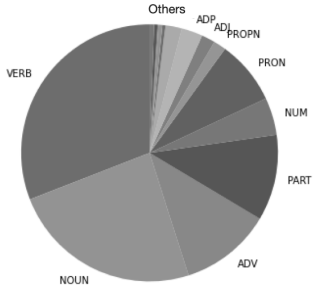

Fig. 10 POS distribution in the reference translations on CSL Daily. VERB and NOUN constitute the largest portion, followed by ADV, PART, PRON and NUM. Other POS tags such as PROPN, ADJ and ADP only takes a small part of all the tokens.

POS Recall

We further group tokens by their POS tags and compute the POS recall, which is calculated by the number of correct predictions of a certain POS divided by the number of total appearance of that POS in the reference gloss or text translation. We plot the proportion of different POS in Fig. 10 and report the POS recall in Fig. 11. Our findings are summarized as follows:

- The overall recall on S2G is higher than other systems.
- The recall distribution on G2T is similar to S2G2T and S2T.
- S2G is most confident on recognizing SCONJ (e.g. “although”, “because”, “until”), while other systems struggle most at translating SCONJ.
- CCONJ (e.g. “and”, “for”, “yet”) and NUM are challenging for all the systems.
- Though having appeared frequently in the training set (Fig. 10), ADV and PART (“up”, “out”, “off”) are not translated properly most of the time.
- VERB and NOUN actually account for a large percentage of errors given their high frequency in the training set, and relatively low recall.

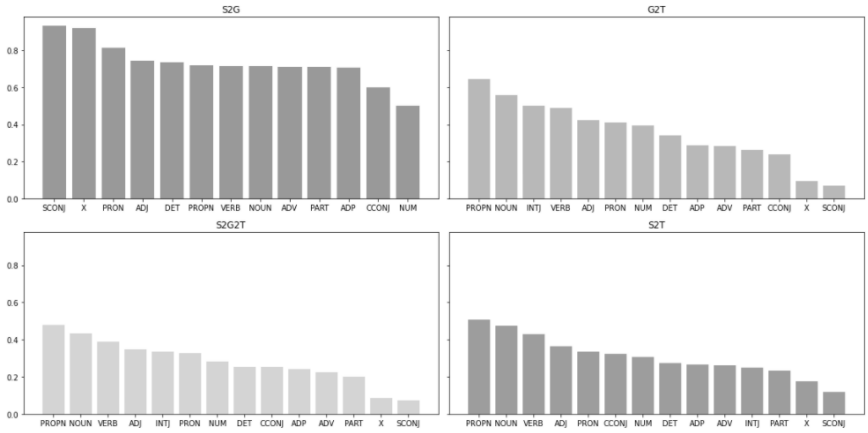

Fig. 11 Recall of POS on CSL Daily. Recall is calculated as the number of correct predictions of a certain POS tag divided by the number of total appearance of that POS in the reference gloss or text translation.

- The rank of the POS-level translation performance is roughly G2T > S2T > S2G2T, which is consistent with the sentence-level rank (Tab. 2).

3.2 Copy-Paste Effect

Glosses and their spoken language counterparts share a large proportion of vocabulary. On CSL Daily, 60.4% tokens in reference text translation also appear in the gloss input. We group text translation tokens by whether they have appearance in the gloss input and calculate the token recall in each group. As shown in Tab. 6, there is a wide gap between the performance on the two groups. G2T, S2G2T and S2T are all efficient at copy-pasting words from the input to the output, but are incompetent at filling in the words that are missing in the input. We call this phenomenon as "copy-paste effect".

Table 6 Recall for tokens that have or have not appeared in the reference gloss.

	G2T	**S2G2T**	**S2T**
tokens in gloss	88.2%	78.1%	75.9%
tokens not in gloss	31.4%	32.1%	28.1%

3.3 Case Studies

We present cases of model outputs that exhibit some of the typical errors in Fig. 12.

Group 1. Copy-paste error. G2T someimes makes mistakes by (1) simply copying a word from gloss without adapting it to the context, or (2) failing to copy and fit the right word into the translation. The example on the left falls into situation (1). G2T copy-pastes the verb "violate" from gloss, while the true translation should instead be the noun "violation". The example on the right demonstrates situation (2). The word "recommend" is falsely discarded by G2T.

Group 2. Compound translation error. As in English, sign languages can form compounds by signing multiple signs in sequence, while the gloss is not necessarily part of the compound in the corresponding text translation. As in the two examples in Group 2, the CSL gloss "happy" and "garden" represent "amusement part" in Chinese, and the CSL gloss "eat" and "good" represent "gourmet" in Chinese. Models in both of the examples struggle at recognizing the compounds.

Group 3. Error propagation from S2G to S2T and S2G2T. Some of the translation errors can be traced back to the malfunction of S2G. In the first example, "taking pictures" is wrongly recognized as "magnetic resonance imaging" (MRI) by S2G, after which the error is passed to S2T and S2G2T, even though "MRI" does not fit in the context. The situation in the second example is a bit different. Although "to treat" is mistakenly recognized as "to shoot" by S2G, S2T fixes the error by the help of G2T, while S2G2T still makes the same mistake.

4 Summary

In this chapter, we identify sign language gloss translation as a low-resource machine translation problem and explore the commonly used approaches to try to improve the performance of gloss translation (Sect. 2). Hyperparameter search (Sect. 2.3) and pretrained multilingual models (Sect. 2.4) both show a great potential, while data augmentation such as rule-based (Sect. 2.5) and back translation (sect. 2.6) might be the most effective when employed with the combination of data selection and curriculum learning (Sect. 2.7). We then view gloss-to-text translation (G2T) as a part of the big picture of SLMT (Sect. 3). We show that the improvement on G2T can be passed to or even amplified in both the end-to-end system (S2T) and the cascaded system (S2G2T). Finally, we conduct error analyses on the model outputs and find that SLMT struggle at translating certain types of POS and has the issue of copy-paste effect.

Fig. 12 Examples of model outputs. **Gloss** is the reference gloss, and **Text** is the reference text translation. **G2T** is the text output of G2T with reference gloss as input. **S2G** are **S2T** are the output of S2G and S2T, which takes the sign videos as input. **S2G2T** is the output of S2G2T with the output of S2G as input.

References

1. Moryossef, A., Yin, K., Neubig, G., Goldberg, Y. (2021). Data augmentation for sign language gloss translation. arXiv preprint arXiv:2105.07476.
2. Camgoz, N. C., Hadfield, S., Koller, O., Ney, H., Bowden, R. (2018). Neural sign language translation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7784-7793).
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. , Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
4. Haddow, B., Bawden, R., Barone, A. V. M., Helcl, J., Birch, A. (2022). Survey of low-resource machine translation. Computational Linguistics, 48(3), 673-732.
5. Zhou, H., Zhou, W., Qi, W., Pu, J., Li, H. (2021). Improving sign language translation with monolingual data by sign back-translation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1316-1325).
6. Sennrich, R., Zhang, B. (2019). Revisiting low-resource neural machine translation: A case study. arXiv preprint arXiv:1905.11901.
7. Zhang, X., Duh, K. (2020). Reproducible and efficient benchmarks for hyperparameter optimization of neural machine translation systems. Transactions of the Association for Computational Linguistics, 8, 393-408.
8. Klein, A., Hutter, F. (2019). Tabular benchmarks for joint architecture and hyperparameter optimization. arXiv preprint arXiv:1905.04970.
9. Ding, S., Renduchintala, A., Duh, K. (2019). A call for prudent choice of subword merge operations in neural machine translation. arXiv preprint arXiv:1905.10453.

10. Koehn, P. (2020). Neural machine translation. Cambridge University Press.
11. Dodge, J., Ilharco, G., Schwartz, R., Farhadi, A., Hajishirzi, H., Smith, N. (2020). Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. arXiv preprint arXiv:2002.06305.
12. Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
13. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S. R. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461.
14. Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S. (2019). Superglue: A stickier benchmark for general-purpose language understanding systems. Advances in neural information processing systems, 32.
15. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I. (2018). Improving language understanding by generative pre-training.
16. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. Advances in neural information processing systems, 32.
17. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer L., Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.
18. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou Y., Li W., Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res., 21(140), 1-67.
19. Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis M., Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. Transactions of the Association for Computational Linguistics, 8, 726-742.
20. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov V., Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461.
21. Chen, Y., Wei, F., Sun, X., Wu, Z., Lin, S. (2022). A simple multi-modality transfer learning baseline for sign language translation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5120-5130).
22. Moore, R. C., Lewis, W. (2010, July). Intelligent selection of language model training data. In Proceedings of the ACL 2010 conference short papers (pp. 220-224).
23. Translation Task - EMNLP sixth Conference on Machine Translation. (2021). https://www.statmt.org/wmt21/translation-task.html
24. Li, Y., Su, H., Shen, X., Li, W., Cao, Z., Niu, S. (2017). Dailydialog: A manually labelled multi-turn dialogue dataset. arXiv preprint arXiv:1710.03957.
25. Wang, Y., Ke, P., Zheng, Y., Huang, K., Jiang, Y., Zhu, X., Huang, M. (2020, October). A large-scale chinese short-text conversation dataset. In CCF International Conference on Natural Language Processing and Chinese Computing (pp. 91-103). Springer, Cham.
26. Zhang, X., Kumar, G., Khayrallah, H., Murray, K., Gwinnup, J., Martindale, M. J., McNamee P, Duh K, Carpuat, M. (2018). An empirical exploration of curriculum learning for neural machine translation. arXiv preprint arXiv:1811.00739.
27. Zhang, X., Shapiro, P., Kumar, G., McNamee, P., Carpuat, M., Duh, K. (2019). Curriculum learning for domain adaptation in neural machine translation. arXiv preprint arXiv:1905.05816.