DGS Glosses to German Text Translation Aided by Temporal Data

Gijs Thissen
STUDENT NUMBER: 2024993

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF SCIENCE IN COGNITIVE SCIENCE & ARTIFICIAL INTELLIGENCE
DEPARTMENT OF COGNITIVE SCIENCE & ARTIFICIAL INTELLIGENCE
SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
TILBURG UNIVERSITY

Thesis committee:

Dr. Dimitar Shterionov
Dr. Sharon Ong

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science & Artificial Intelligence
Tilburg, The Netherlands
May 2021

Preface

Foremost, I would like to express my gratitude towards all persons that helped me during the creation of this thesis. From helping me with finding the topic to helping me correct my mistakes while programming the algorithms used in this thesis.

I would like to especially thank my research supervisor **Dr. Dimitar Shterionov** (Assistant Professor and Researcher at Tilburg University) for the guidance given by helping me both with the research for this topic and correcting my many mistakes while coding.

I am grateful to my fellow Thesis colleagues **Stan van der Vossen** (CSAI-Student), **Jakob E. Hauser** (CSAI-Student), **Stefano Scola** (CSAI-Student), and **Codrin Mironiuc** (CSAI-Student) for the advice, translating sporadic German to English, and guidance given towards solving the problems I faced during the creation of this Thesis.

Lastly, even though they are unaware of me writing this Thesis, I would like to express my gratitude towards all the researchers that aided in the creation of the DGS-Korpus and especially for making it available to the online public.

Gijs Thissen
Tilburg, May 2021

DGS Glosses to German Text Translation Aided by Temporal Data

Gijs Thissen

Sign languages are languages that have unique grammar, words, and manners of speaking. Therefore they may be completely distinct from the regional equivalent spoken language. Most of the time these live in parallel worlds, with the deaf on one side and the hearing on another. When these worlds clash, confusion arises because of the dissimilarity between the two since they might not stem from the same language family. In recent years the field of neural machine translation has grown exponentially, with the invention of the transformer architecture increasing accuracy in machine translation software. German Sign Language (DGS) and German are two distinct languages and are therefore subject to translation possibilities. The recent approaches mainly focus on (German) text to (DGS) glosses. However, there has been a lack of a glosses-to-text translation system. The focus of this paper is to find the best approach by adding temporal, vocal, or combined data to the glosses of the DGS Public Korpus. The results show that adding these extra tokens to the data results in a less accurate translation across all models. The neural machine translation system with an input of nothing but glosses [BLEU 3.69, TER 0.960] outperforms the more complex models [average BLEU 2.19, average TER 0.981]. It is concluded that adding temporal, vocal or both in the data without the use of standardization of the tokens aids in the emergence of more rare and unique words resulting in the decrease in accuracy of the model.

1. Introduction

Contrary to popular belief, sign language and spoken language are not the same. In fact, sign language has its own words and grammar (Camgoz et al., 2018; Stokoe Jr, 2005) that are completely separate from its regional spoken counterpart. The earliest attested use of sign language can be found in Plato's Cratylus (Sedley, 2003). The hearing impaired, users of sign language, use signs as their day to day communication (Perniss, 2007a). This is done by the use of spatial-temporal gestures to communicate pieces of information. However, communication between the signers and non-signers can prove to be difficult (Meadow et al., 1981) when both of them do not know the other's language.

1.1 Spoken Language of the Deaf and Hard of Hearing

When a deaf person signs, spoken words can be heard. In the field of sign language research, these spoken language utterances performed by deaf individuals are referred to as mouthings (Konrad et al., 2020; Braem & Brentari, 2001). The concept of mouthings is quite common among non-deaf individuals. It is often referred to as intra-sentential code-mixing (Paradis et al., 2000) and is fairly prominent among bilingual non-deaf children. However, bilingualism does not limit itself to solely hearing individuals.

Many deaf persons are, in fact, bilingual. Only 8% of deaf children are born to two deaf parents. Consequently, these children tend to "speak" both sign language and spoken language (Perniss, 2007a). Research suggests that in adults, this code-mixing has become a structural part of their linguistic abilities and plays an important role in the meaning of a sign. (Pfaff, 1979; Konrad et al., 2020). It is, however, not consistent among sign languages. Signers of American Sign Language tend to use less mouthing when compared to signers of German Sign Language.

1.2 Sign Translation

When faced with situations where proper communication is key, written text is a solution to circumvent ambiguity. In the case of sign language, linguists have developed conventions for translating sign language into text (Konrad et al., 2020; Aouiti et al., 2015; Porta et al., 2014). This is done by denoting a single gesture as a word. Around this word, several glosses are positioned. Glosses are brief annotations used by linguists to denote an explanation of a word. For example, I LIKE [negative] COWS would translate to I don't like cows. Translating glosses to normal text is a tedious job that is exclusively done by specialised linguists. Preferably this task would be able to be automated. However, for this goal to be achieved, there is a need for a corpus to be machine-readable. Machine readability is achieved by making use of consistent sign annotations and tags (Johnstonet al. , 2008).

To solve this problem, end-to-end machine translation models have been developed to recognize gestures and translate gestures directly to text (De Coster et al., 2020). Sign recognition systems work by isolating different signs from each other, which has proven to be difficult due to the existence of a transitional period between signs (Konrad et al., 2020) (See Section 3.4 about Segmentation for more details). While this approach has shown promising results it ignores the underlying linguistical aspects of sign language (Camgoz et al., 2018; Perniss, 2007a). This paper will investigate whether or not adding temporal data improves machine translation systems for language glosses to text.

1.3 Research Question

To what extent does temporal data aid in the performance of a machine translation system for sign language glosses to text?
The research question is broken down into the following sub-questions:

- **How to introduce temporal data to the Machine Translation system?**
  The baseline system does not have temporal data added to it. This sub-question is related to how to add temporal data to the machine translation system.
- **To what extent does temporal data aid a Machine Translation system when compared to a baseline system?**
  The chosen dataset (Konrad et al., 2020a) contains both the glosses (i.e.: TO-HAVE-BSL1, I1^, BUTTER1A) and the timestamps next to them to indicate when and how long these signs have taken place. This sub-question is related to combining this temporal data with the baseline system. Both the baseline and the temporal experiments are described in the methods section.

- **To what extent does vocal data aid a Machine Translation system when compared to a baseline system?**
  The dataset contains both glosses on sign language and data on what the subject said during the interview (vocal). This sub-question is related to:
  1. Combining the additional vocal data with the baseline system.
  2. Combining the additional vocal data with the temporal data and the baseline system.
  3. Measuring to what extent does adding extra tokens have an effect on the performance of the Machine Translation system.

  .

1.4 Findings

The results of the conducted research show that combining words and temporal features does not increase the accuracy of the model. Instead, combining these features decreases the BLEU score of the translated text by as much as 22%. Even though prior research showed that mouthings are intrinsically part of a signer's linguistical ability (Pfaff, 1979; Konrad et al., 2020) the results show a 45% decrease in BLEU score when paired with words and fed into an NMT system. Combining all the data results in a BLEU score loss of 50% on the translated text.

2. Related Work

Research on parallel translation of spoken language is widely attested. Sign language translation, however, is a relatively small field with little appropriate datasets or methods (Camgoz et al., 2021; Bragg et al., 2019).

2.1 Hidden Markov Models

In the late 1990s, the field of Sign Language translation wasn't much of a separate field, the research that did exist was mainly focused on recognizing sign language. The focus of Sign Language Recognition mainly relied on HMMs (Starner & Pentland, 1997; Liang & Ouhyoung, 1996; Vogler & Metaxas, 1999; Holden et al., 2005). A gesture would be decomposed into a sequence of postures. These postures would then be recognised by the HMM as a gesture. For the system to be able to track a gesture the user would be wearing an annotation glove. For example, coloured rings would be worn around each finger. This approach changed in 1996 by using the colour of the participants' skin to segment from the background by calibrating the system on the participants noses (Starner et al., 1998). This proved to be problematic, however, since the hands move at a different pace compared to the nose the system would occasionally filter out the nose (Starner et al., 1998). In 2005, an end-to-end pipeline, called the Sign2 Conversion System (Glenn et al., 2005), was created. However, it could only be used for ASL fingerspelling excluding any non-alphabetical ASL signs and therefore limiting the system greatly by excluding essential linguistical aspects of the American Sign Language.

2.2 Deep Learning

Using neural networks in sign recognition was among the first methods proposed at the onset of the field (Murakami & Taguchi, 1991; Fels & Hinton, 1993), however, the approach was overshadowed due to the success of HMM's in the 1990s (Cooper et al.,

2011). The use of neural networks became more prominent in the late 2000s (Parton, 2006). During this time several new international projects were set up to create sign language recognition systems for local sign languages (Admasu & Raimond, 2010; Akmeliawati et al., 2007; Kiani Sarkaleh et al., 2009; Dias et al., 2009; Maraqa & Abu-Zaiter, 2008).

Within recent times Deep Learning (LeCun et al., 2015) has gained popularity giving rise to new models. This enables the use of 3-dimensional convolutional neural networks. While previous recognition systems employed sign language recognition on 2-dimensional images, due to having no temporal dimension in their kernel this would result in a 2-dimensional output (Tran et al., 2015). The main difference between the old systems and the new is the 3-dimensional convolutional neural network (CNN) that can extract discriminative spatial-temporal features (Huang et al., 2015).

2.3 Continuous Sign Language Recognition

In mid 2010s the field has morphed into continuous sign language recognition (CSLR) (Koller et al., 2015). Where the older SLR systems were limited to the use of isolated gestures (Cooper et al., 2011), through the use of (Long Short Term Memory) LSTM models (Hochreiter & Schmidhuber, 1997) the field was able to create a CSLR system that recognizes gestures as a sequence of interconnected sub-units (Mittal et al., 2019). This proved to be especially difficult since there are no clear boundaries between signs hands move from because signs do not simply stop but rather move in transitional movements (See Section 3.4 for details about Segmentation). An architecture that improves on this, even more, is the Transformer architecture (Vaswani et al., 2017) improving the WMT 2014 English-to-German translation task by 2 BLEU score (see Section 4.5 for details about BLEU) compared to older architectures. Consequently this architecture was applied to the field of SLR where Transformers have been proven to outperform LSTMs in SLR experiments (De Coster et al., 2020; Camgoz et al., 2020).

2.4 Sign Language Translation

The field of sign language recognition has continuously posed the problem of translating sign language as a purely symbolic one (Camgoz et al., 2018). However, sign language is its own language with its own distinctive grammar (Perniss, 2007a; Camgoz et al., 2018; Camgoz et al., 2020), consequently by ignoring the grammar and linguistic properties of a sign language important information is lost. The current state-of-the-art solves this issue by using sequence-to-sequence based deep learning models. The SLR translation as performed by Koller et al., 2015 is still performed, however, it is enhanced by embedding the glosses into the system (Camgoz et al., 2018). Due to the lack of prior research into the topic of Sign Language Translation, especially regarding Neural Machine Translation, many tokenization methods have yet to be explored. In the NMT system, the temporal features are added, however, contrary to the signs and their respective glosses, no temporal data is embedded. Research has also shown that the mouthing a deaf person performs is of importance for the meaning of the sign that is being performed at the same time (Konrad et al., 2020).

This thesis focuses on to what extent the addition of temporal or vocal tokens, aids in the performance of a gloss to text NMT system. The system will use the Transformer architecture in combination with source word features (García-Martínez et al., 2016) that correspond to either the temporal, vocal or combined glosses. The

public DGS corpus (Konrad et al., 2020a) will be used due to the presence of both vocal and temporal tiers in the corpus. It is hypothesised that adding temporal, vocal, or either will progressively strengthen the NMT-system's accuracy while reducing the errors it makes in the translation.

3. Data

In the following section the source and distribution of the data for the NMT system will be discussed as well as the method in which the dataset was collected (IDGS, 2020). Furthermore, the conventions by which the corpus was created is presented and explained in detail.

3.1 Source

The data consists of 405 EAF-files (ELAN Annotation Files) gathered by the Institute for German Sign Language and Communication of the Deaf at Hamburg University (Prillwitz et al., 2008). These files consist of a total of 50 hours of annotated recordings spanning a wide range of narrations regarding the cultural aspects of the deaf community. The interviews were conducted using a peer-to-peer procedure, where participant change roles according to the conversation (Prillwitz et al., 2008). During a discussion 2 German Sign Language or Deutsche Gebärdensprache (DGS) signers conversed. Each conversation consisted of a standardized interview covering linguistical and social data (Hanke et al., 2009). Following this conversation, a spontaneous conversation on a given topic was held while the participants were encouraged to use as much basic vocabulary as possible (Hanke et al., 2009). In the EAF-files the signers are annotated as either Speaker A or Speaker B. The data was collected between January 2010 and December 2011 by videotaping 330 participants from all 16 Federated States (Bundesländer). The recordings were conducted using a mobile field lab in; areas with a relatively high deaf population density (deaf schools, deaf centres, and deaf institutes) (IDGS, 2020) (Hanke et al., 2009), the catchment areas of the former Schools for the Deaf (IDGS, 2020; Prillwitz et al., 2008), areas which were easy to reach from surrounding rural communities (IDGS, 2020; Hanke et al., 2009), and finally areas that were suspected of having a distinct dialect. (IDGS, 2020; Prillwitz et al., 2008)

The data was collected by representatives of the local deaf community to take into account the regional varieties of DGS (Hanke et al., 2009). As can be seen in Figure 1 the data was sampled from participants across different age groups (IDGS, 2020).

3.2 ELAN

The conversation conducted between the representatives of the deaf community has been annotated and stored in EAF-files. These EAF-files can be shown and edited using the ELAN software (Crasborn & Sloetjes, 2008). ELAN is a multimedia annotation software developed by (Nijmegen: Max Planck Institute for Psycholinguistics, 2020) to assist in Linguistical studies, language conservation, and sign language research (Brugman et al., 2004). The software can be used to add annotations to video and/or audio recordings. An annotation can be a sentence, word, or in the case of sign language, a gloss. Using ELAN multiple annotations can be created that are sorted into tiers (Crasborn & Sloetjes, 2008).

Figure 1
Showing the percentage of participants per age, divided into both the Male (shown in Purple) and Female (shown in Red) gender, that participated in the creation of the DGS corpus (both annotated and non-annotated). (IDGS, 2020)

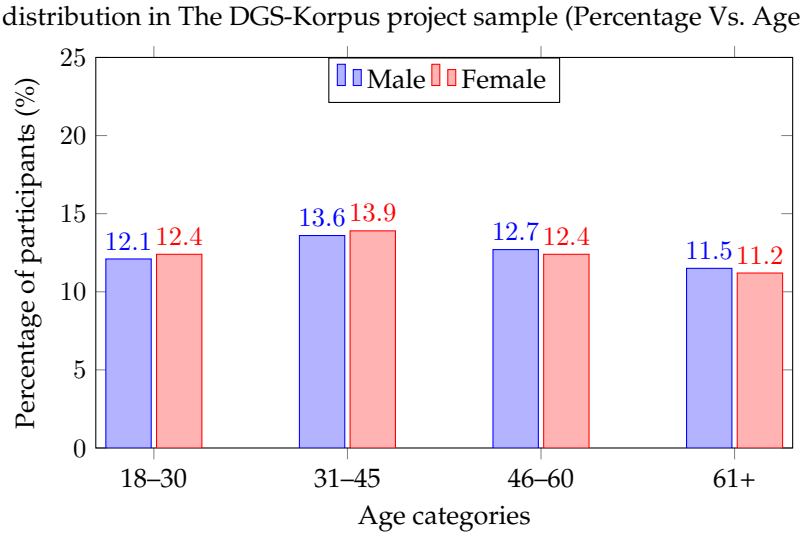
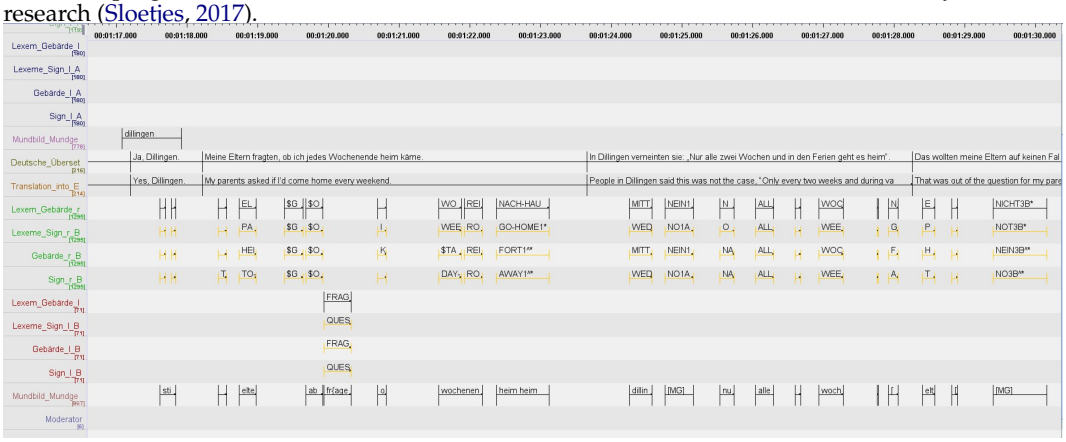


Figure 2
Showing the structure of one of the EAF (English Annotated File)-files (Konrad et al., 2020b) inside the program ELAN. ELAN is a multimedia annotation tool used for multi-modality research (Sloetjes, 2017).



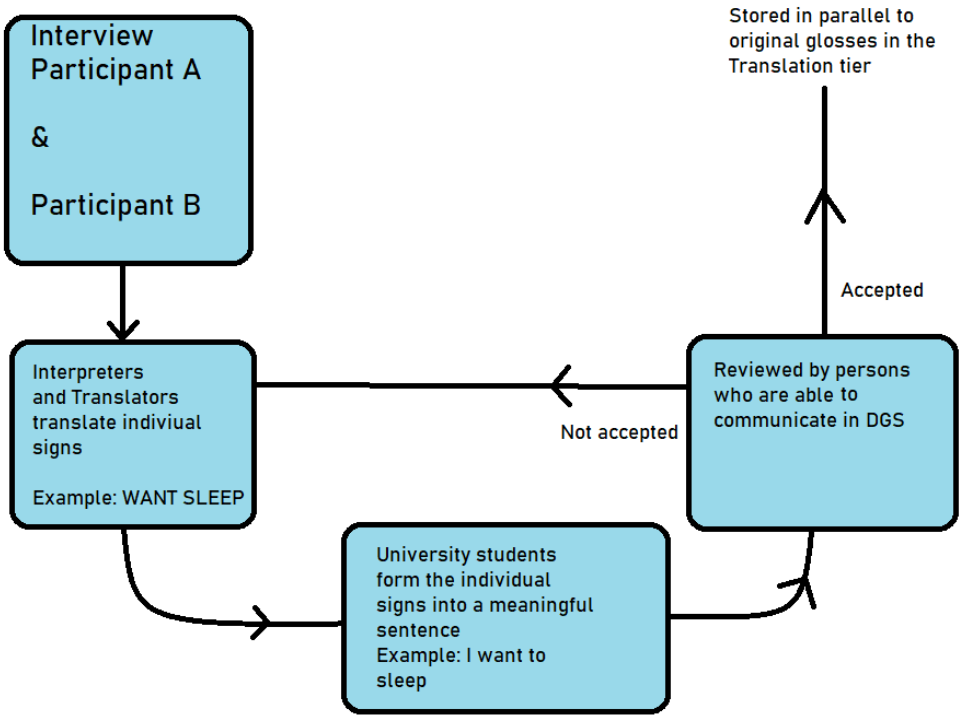
3.3 Initial Translation

The annotations are divided into tiers are presented across a timeline. As can be seen in (Figure 2) these files are divided into 11 different tiers, presented in both English and German due to accessibility (Konrad et al., 2020). Take "Lexem_Gebärde_r_A" for example, this tier is translated into English as "Lexeme_Sign_r_A". The tiers present in the EAF-files, translated into English, are: "Time", "Sign_l_B", "Sign_r_B", "Lexeme_Sign_l_B", "Lexeme_Sign_r_B", "Translation_into_English_B",

"Sign_l_A", "Sign_r_A", "Lexeme_Sign_l_A", "Lexeme_Sign_r_A", and "Translation_into_English_A" (Sloetjes, 2017). The goal of this research is to investigate to what extent temporal data aids in the performance of a machine translation system for sign language glosses to text. For this purpose, both the German Sign Language and the German Language were chosen respectively. Since the target language is German, since the corpus originated at the university of Hamburg (IDGS, 2020), it was decided to use the original glosses and translations that were available in the corpus.

The initial translation of the glosses in the data set (i.e.: Sign_l_A to Translation_into_English) was conducted by contracted sign language translators and interpreters (Konrad et al., 2020). These researchers translated the data set word-for-word. Consequently, university students created coherent meaningful sentence like structures. Lastly, these sentences were fed back into the system until the proper meaning was determined by the DGS experts (Figure 3) (Konrad et al., 2020).

Figure 3
Showing the translation pipeline of the initial translation phase as conducted by the Institute for German Sign Language and Communication of the Deaf at Hamburg University. (Konrad et al., 2020)



3.4 Segmentation

As with spoken languages, sign languages tend not to have a naturally occurring white space character (Hanke et al., 2019; Konrad et al., 2020). Thus it is difficult to determine where a sign begins and where a sign ends. In this situation, the segmentator has two options: either add a gap or none at all. Unlike spoken languages, when a participant is

signing two words, take TREE (rest your right forearm upon your left palm and twist) (Perniss, 2007b) and COW (make two horns using your thumb and little finger on top of your head), for example. There is a transitional period where the participant moves his or her arm from one sign to the next. When adding a gap between the signs, the segmentator determines that the transitional movements are not part of the token's form (Konrad et al., 2020). Taking this into account researchers at the University of Hamburg have decided that for the creation of the DGS corpus gaps will be added. The direct result of this decision is located in the EAF files, where there is temporal data assigned to the white spaces.

3.5 Lemmatisation and Gloss conventions

When creating a corpus there is an intrinsic need to have conventions in place to make sure a uniform dataset is created (Konrad et al., 2020). Therefore it is necessary to employ linguistics to standardise glosses using gloss conventions (Kristoffersen et al., 2016):

- In the field of Linguistics or more specifically in the field of lexicography, a lexical item forms the basic element of the lexicon of a language. Lexical signs are treated as items, that is as units of their respective sign language that would be found in the dictionary (Konrad et al., 2020). When a deaf person signs SQUARE1^it may mean different things such as a map, a recipe, or a page. In DGS (Perniss, 2007b) and several other European Sign Languages signs are iconically motivated (Pietrandrea, 2002; Oomen, 2017), meaning that there is a similarity between the form of the sign and the meaning of the sign. In the DGS corpus, type glosses are given an indication of iconic value by using a circumflex at the end: "SQUARE1^". Examples of child types to the SQUARE1^parent type are: FORM1, MIRROR2, and PAPER4, these and all other non-circumflex glosses are subtypes. The numbers denote different lexical variants.
- Fully iconically motivated signs, also known as poly-morphemic signs, are denoted as productive signs in contrast to lexical signs that denote a object in nature instead of meaning. Productive signs, therefore, have a $PROD token (Konrad et al., 2020)
- Due to the anonymisation laws present in Germany, where this corpus was created, all names are replaced by "$NAME". Except when it concerns a famous person the $NAME gloss is followed by the person's name. (Konrad et al., 2020)
- Foreign language elements are represented using the INTS token, for example, when signing the English word Germany instead of "Deutschland" it has been written as GERMANY-INST1. (Kellett & Ochse, 2008; Konrad et al., 2020)
- In German Sign Language it is conventional to use a one-handed manual alphabet, for example when spelling out someone's name. For these situations, the $ALPHA token is used. Numbers are presented similarly by using $NUM. (Konrad et al., 2020)

Taking into account these points a uniform corpus was created with annotated signs presented on a timeline. An example of an annotated DGS sentence can be seen in Figure 4.

Figure 4
Showing an example sentence of the annotated corpus in DGS glosses. The German translation of this gloss sentence is: "Und dann fällt es einem wieder auf, dass der andere noch Fehler macht" (Konrad et al., 2020a). (English translation: And then one notices again, that the other (person) is still making mistakes). 1. A token denoting a gesture. 2. The sign annotated as NM has iconic meaning. 3. One of several lexical variants of the lexical unit FEHLER.

4. Methods

Since the data was created under specific gloss conventions (Konrad et al., 2020) it is difficult to build a MT system with. Furthermore, the data was split up into multiple separate files and it was therefore necessary for these files to be ordered and combined. This section will show the various techniques that were employed in this process.

4.1 Experiments

To answer the (sub-) research questions, experiments are needed. Described below are 4 experiments that have been designed to test the hypotheses, they differ mainly in the data structure. A good overview of their differences can be found in (Figure 5).

4.1.1 Baseline. Other experiments will be compared to the baseline. This model lacks any externally added tokens. The data consists of a sequence of glosses as the source and their German translations as the target (Source: PRIVAT1A* FAMILIE1 Target: Privates oder Familie? (Konrad et al., 2020a)). The sign language glosses are without any transitional periods (See the Lemmatisation and Gloss conventions section in Data) and adhere to the standard gloss conventions.

4.1.2 Temporal. One of the sub-goals of this thesis is to, inspect or identify to what extent temporal data aids in the performance of machine translation systems for sign language glosses to text. This will be tested in the Temporal experiment. The data in the temporal experiment consists of the glosses formed into a sentence with tokens added to each word separated by the "|" character, e.g.: FAMILIE|120. The number behind the | character represents the total amount of milliseconds it took for a sign to be signed, in the given example this would mean that the sign for FAMILIE (English: Family) would have taken 120 ms to be made. As mentioned before these sentences are without the transitional periods from one sign to another (See Section 3.5). The "|" sign is a library standard sign to split Word features from the actual words (Klein et al., 2017). The temporal data files consist of glosses formed into a sentence with added temporal data and their German translation as the target (Source: PRIVAT1A*|340 FAMILIE1|460 Target: Privates oder Familie? (Konrad et al., 2020a))

4.1.3 Vocal. Another sub-goal of this thesis is to investigate to what extent adding extra embeddings aids in the performance of machine translation systems for sign language glosses to text. This will be tested in both the Vocal and the Combined experiment. The data in the vocal experiment consists of glosses formed into a sentence with tokens added to each word that represent the spoken words uttered by the participant while performing this sign. Research has shown that the mouthing a deaf-person performs is of importance for the meaning of the sign that is being performed at the same time (Konrad et al., 2020). The spoken word and the gloss will be separated by the "|" character as per library standard (Klein et al., 2017). When using the OpenNMT library (Klein et al., 2017) it is necessary that all source word features have the same amount of features. Thus when a sign does not have an accompanied mouthing a "nan" will be added instead of the spoken word. This is done to represent the empty space created by the empty row. The vocal data files consist of glosses formed into a sentence with added vocal data and their German translation as the target (Source: PRIVAT1A* | privat FAMILIE1 | familie Target: Privates oder Familie?). It is important to note that all vocal word features are denoted solely using lowercase characters.

4.1.4 Combined. Research has shown that talking speed is affected by emotions (Kshirsagar, 2002). The combination experiment is an experiment that combines the previously mentioned temporal and vocal features into one model. The data will be separated by a "|" and be processed in a GLOSS | vocal | TIME format. The same conditions apply to the combined model as applied to the vocal and temporal model. The combined data files consist of glosses formed into a sentence with added vocal and temporal data and their German translation as the target (Source: PRIVAT1A* | privat | 340 FAMILIE1 | familie | 460 Target: Privates oder Familie? (Konrad et al., 2020a)).

Figure 5
Showing the source file textual comparison between the Baseline, Temporal, Vocal, and Combined experiments. In the temporal experiment, the concept of time is added (in ms). In the vocal experiment, the concept of spoken words is added (in lowercase letters). In the combined experiment these are combined into one.

Baseline: PRIVAT1A* FAMILIE1

Temporal: PRIVAT1A*|340 FAMILIE1|460

Vocal: PRIVAT1A*|privat FAMILIE1|familie

Combined: PRIVAT1A*|privat|340 FAMILIE1|familie|460

4.2 Pre-Processing

4.2.1 EAF to CSV. Using the ELAN software 7 of the 11 tiers were selected and converted into a CSV file. This creates 2 distinct CSV files that were consequently merged (section 1). Due to having merged the files, there are no longer multiple persons in the data set, instead, the data set is transformed into a contextless database. Due to being able to access the timeline in ELAN (Nijmegen: Max Planck Institute for Psycholinguistics, 2020), it is possible to extract the temporal features of a given annotation. As mentioned before the annotators decided to add gaps between the signs (Hanke et al., 2019; Konrad

et al., 2020), this has been carried over into the EAF-files, resulting in empty rows inside the CSV file. These need to be removed since the program automatically puts white spaces between words instead of pasting them into one giant string (section 1). Failing to do so would result in temporal features being assigned to empty strings (A temporal-annotated sentence without removal of prior white spaces would result in $GEST-NM^|430 |230 FEHLER1*|450 |400 $GEST^|400).

4.2.2 Data splitting. As has been described 4 experiments are to be performed: Sign, Temporal, Vocal, and Combined. For each of these experiments, two files are needed: a Source file (.src), and a Target file (.trg). These files are fully extracted from the data frame. By isolating the sentences in the data frame it is possible to create a separate sentence set for every sentence. This will result in 5 files: normal.en, vocal.en, times.en, combined.en and sentences.nl. "sentences.nl" is the target file for all the experiments and is therefore uniform.

To tokenize and split the data the `train_test_dev.py` (Shterionov, 2020) script was used to both tokenize and split the data into a training, a validation, and a testing file. Since the data set only has 60000 sentences it was decided to split the data into only 500 sentences for validation and testing each while the remaining 59000 sentences were used for the training file.

An unwanted side-effect of the `train_test_dev.py` (Shterionov, 2020) script is that it adds white spaces between the individual sentences in both the (train, test, dev) source files as the target files. This results in the Machine Translation system "assigning" sentences to these white spaces when training, resulting in a normal German sentence within a normal German sentence. For this reason all files are passed through the `remove_whitespace.py` script (section 1) which will strip away all empty lines.

4.2.3 BPE. Byte Pair Encoding (BPE) is a general-purpose data compression algorithm (Gage, 1994) which has been employed for MT to reduce the OOV instances (Sennrich et al., 2015). The way it works is by replacing the most frequent pair of bytes in a sequence with unused bytes, in the case of the OpenNMT Byte Pair Encoding (Klein et al., 2017) scripts the unused bytes were denoted in the data files as "@@ ". The advantages of using the BPE algorithm are:

1. **Memory**. In the case of German, many words start with the unit "auf", by encoding this unit by only using 1 symbol, the system only encounters one symbol instead of 3. (Gage, 1994)
2. **Out-of-vocabulary words (OOV)**. Taking into account the previous example, since the sub-unit "auf" has been replaced by 1 symbol encountering it will be familiar. This is because BPE allows for the encoding of rare words and will not introduce any unrecognised tokens. When the system thenceforth encounters words containing the sub-unit it will be more familiar rather than it would be in the case without the sub-unit. (Gage, 1994; Sennrich et al., 2015)

Before being passed to the pre-processing and training, the data, with the exception of the test.trg-file, since after translation the output file will be passed through an un-encryption algorithm resulting in normal text, were passed through a modified form of the BPE algorithm. In normal circumstances when using BPE, it caused no problems, however, when applying BPE to data containing temporal word features it would split

the word on the "|" character. The modified algorithm first double-checks whether or not there is a temporal feature added to the word and if there is will join the word and its respective temporal feature together (section 1). Applying `remove_whitespace.py` on the data files significantly increases the BLEU-score for the Baseline model, while the old Baseline (pre-BPE, post-removal) had a BLEU score of 2.15, the new Baseline (post-BPE, post-removal) had a BLEU-score of 3.69. It was therefore decided to use the Byte Pair Encoding algorithm to compress the data files.

4.3 OpenNMT

To train the data a Neural Machine Translation library is needed that supports source word features. With these criteria in mind 2 libraries were eventually found: OpenNMT (Klein et al., 2017) and MarianNMT (Junczys-Dowmunt et al., 2018). A main difference between the two remaining libraries is the way source word features can be used. Consider the following sentence: "PRIVAT1A*|340 FAMILIE1", as can be seen, the first word has word features attached, however, the second word does not. While this is not a problem in MarianNMT this would be a cause a problem in OpenNMT since it's source word features need to be consistent. Nonetheless, OpenNMT was chosen as the preferred library because of its compatibility with Microsoft Windows 10.

4.4 Hyperparameters and Architecture

Using OpenNMT (Klein et al., 2017) a Transformer model was trained. The hyperparameters were picked to match the recommended standard transformer hyper-parameters (OpenNMT, 2018) as shown in (Table 1). These settings were chosen to create a model that is able to imitate the WMT2014 German-English (Bojar et al., 2014) results as achieved by the original paper on Transformers (Vaswani et al., 2017). Overfitting on a small data set (the training data consists of 60000 sentences) is a major challenge (Barone et al., 2017), therefore, an early stopping criterion was introduced. If the perplexity and accuracy of the model do not improve during the last 5 validation performances the system will be stopped. Due to the limited capabilities of the system used, even with the use of the BPE algorithm, the batch size had to be decreased from a recommended 4096 to 1024.

5. Evaluation Metrics

5.1 BLEU

BLEU is short for Bilingual Evaluation Understudy and is an evaluation metric for Neural Machine Translation as described in (Papineni et al., 2002). As the name suggests it is an evaluation metric for a parallel bilingual system with a reference (original target file) and a hypothesis (predicted target file). Previously used metrics were human-based and could take weeks or even months to be calculated (Papineni et al., 2002). The idea behind the metric was that the closer a machine translation was to a professional (linguistic) human translation, the better it was. Therefore one has to measure how close the prediction is to the reference translations.

5.1.1 Weighted precision. BLEU uses the weighted precision score (Figure 6). The difference between the modified version and the normal version is that the modified version takes into account the maximum reference count (Chan, 2012). If the "normal"

Table 1
Showing the hyperparameters used for training using the Transformer architecture.

Hyperparameter	Setting	Hyperparameter	Setting
Encoder type	Transformer	Batch size	1024
Decoder type	Transformer	Batch type	tokens
Transformer feed-forward	2048	Normalization	tokens
Layers	6	Warm-up steps	8000
Heads	8	Training steps	20000
RNN size	512	Validation steps	1000
Word embedding size	512	Label smoothing	0.1
Position encoding	True	World size	1
Maximum generator batches	2	GPU rank	0
Dropout probability	0.1	Early stopping	5
Optimization method	adam	Early Stopping criteria	perplexity, accuracy
Adam beta2 hyperparameter	0.998	Max gradient norm	0
Decay Method	noam	Parameters initialized at	0
Learning rate	2	Parameters_init_glorot	True

precision measure was used with multiple references, when comparing the sentence "dog dog dog dog dog dog dog" with "I want a dog" would be able to get a score of close to 1. This is especially true since MT systems tend to over-generate words that seem to fit in (Papineni et al., 2002).

Figure 6
Showing the modified n-gram precision formula. The difference between the modified version and the normal version is that the modified version takes into account the maximum reference count, known as the clipped count. Candidates refers to the translated sentences (Chan, 2012) (Papineni et al., 2002). First compute the N-gram matches. Second add the clipped counts for all translated sentences. Third divide the sum by the total number of n-grams in the reference. (Chan, 2012)

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}.$$

5.1.2 Brevity penalty. Traditionally the BLEU-score was used to calculate the score of a prediction over multiple references. Since these references may or may not have variable length this will negatively affect the recall score (Papineni et al., 2002) (Section 2.2). The proposed solution is the brevity penalty or BP for short as can be seen in (Figure 7). The Brevity penalty high scoring translations must match the references in word order and length. Sentences smaller than the reference sentences are therefore penalized since they are multiplied by a factor < 1.

Figure 7
Showing the complete formula for calculating BLEU (Papineni et al., 2002). In the BP formula, c denotes the total length of the translation corpus, r is the sum of the best match lengths of the translation sentence in the test corpus (Papineni et al., 2002). If the BP value is 1 this means that the translation length and the reference length are equal if this is not the case the second formula will be used. The eventual BLEU-score is the average of the weighted precision scores times the brevity penalty and therefore penalizes shorter sentences.

$$\text{Brevity penalty BP} = \begin{cases} 1 & if\ c > r \\ e^{(1-\frac{r}{c})} & if\ c < r \end{cases}$$

$$\text{BLEU} = \text{BP}^{*}\exp\left(\sum_{n=1}^{N} w_n log(p_n)\right)$$

5.1.3 BLEU-score. The BLEU score ranges from 0 to 1, with 0 being not related to the reference sentence and 1 being identical. As a convention in the field of Neural Machine Translation, the BLEU-score is either multiplied by 10 or 100, for the purposes of this paper the BLEU-score will be multiplied by 100. In this paper, BLEU will be used on only 1 reference since there is only one available per gloss sentence.

5.2 TER

Translation Edit Rate (TER) measures the amount of editing that needs to be done to create an output that exactly matches a reference translation (Snover et al., 2006). There can be multiple TER-scores since they may match one of many references, however, since in this paper only one reference will be used this will not be considered a problem. The higher the TER-score the worse the translation, since the more edits are performed on the total number of edits. The formula to calculate the TER-score is shown in (Figure 8). The number of edits is calculated in two different phases (Shapira & Storer, 2002):

1. A greedy search algorithm is used to find the words that need to be shifted from one place to another.
2. An optimal calculation is made to calculate the smallest amount of remaining edits (insertion, deletion, and substitution) necessary to match the reference.

Figure 8
Showing the formula for calculating the TER score (Snover et al., 2006). The TER-score is calculated by dividing the number of edits by the average number of words in the reference translation. Possible edits are insertion, deletion, substitution, and switching words around. All these edits have an equal cost.

$$\text{TER} = \frac{\text{\# of edits}}{\text{average \# of reference words}}$$

6. Results

Table 2
Best performing models with the accompanying BLEU- and TER-scores. Highlighted in bold are the best results. As can be seen in the table the baseline outperforms all other models.

Word-Form	Models	BLEU	TER
Original	Baseline	**3.69**	**0.960**
	Temporal	2.85	0.971
	Vocal	2.03	1.000
	Combined (Temporal + Vocal)	1.69	0.973

At the beginning of this paper it was hypothesised that with the insertion of temporal data into the baseline data, there would be an improvement of the translation quality. Additionally, it was hypothesised in a sub-hypothesis that the addition of a vocal token would increase the BLEU-score of the model (see Section 4.1.3 on the scientific background). For these reasons 4 experiments were set up, the baseline, temporal (baseline + time), vocal (baseline + vocal), and combined (baseline + vocal + time). These experiments were performed, on each of them, their respective BLEU and TER score were calculated these results are presented in (Table 2). Histogram plots are shown in Figure 9 and 10 to compare the scores to each other and see any possible trends between them.

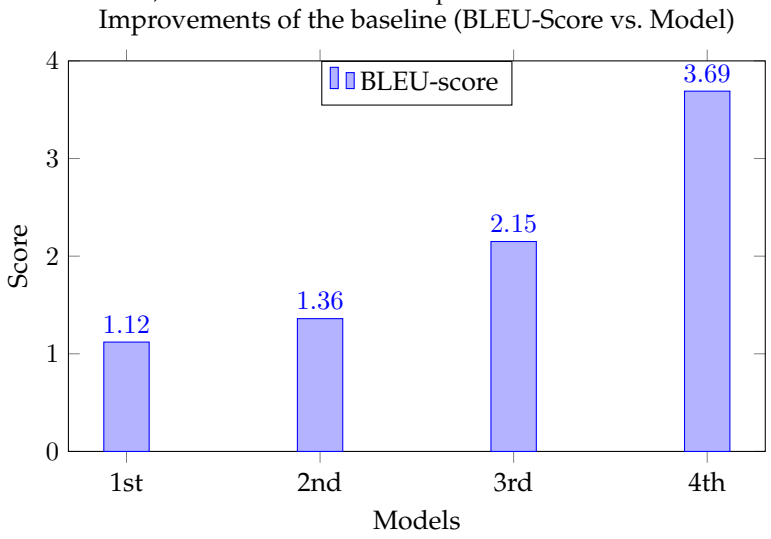
6.1 Baseline fine-tuning

The baseline experiment consisted of 4 sub-experiments, the results are shown in (Figure 9). The first experiment that was run, ran on basic hyperparameters, such as a learning rate of 0.01, however, this architecture was abandoned in favour of mimicking the original transformer architecture. The results improved by 17.6% (1.12 -> 1.36). The fine-tuning of the third experiment had to do with the pre-processing. In both the first and second experiment an oversight was made regarding empty lines in the data files, these would result in sentences being learned for empty source lines. Applying `remove_whitespace.py` on the data files significantly increases the BLEU-score for the Baseline model, while the old Baseline (pre-BPE, pre-removal) had a BLEU score of 1.36 the new Baseline (pre-BPE, post-removal) had a BLEU score of 2.15. The algorithm improved the results by 36.7% (1.36 -> 2.15), it was therefore decided to use the `remove_whitespace.py` on all data files. The fourth and final experiment was tuned by applying the BPE algorithm (Gage, 1994) (see section 4.2.3). The results improved by a total of 41.7% (2.15 -> 3.69). The translated text of the final baseline translation model had a BLEU-score of 3.69 and a TER score of 0.960, an example is shown in (Table 3). The former is an improvement of 69.6% compared to the non-tuned baseline while showing a general trend of improvement across the sub-experiments. When the BLEU-score is 3.69 and the TER score 0.960 are compared to contemporary research they are below average. However, of the 4 conducted experiments the baseline model had the best performance.

Table 3
Showing the translation results of the baseline model, the original input is the input of the glosses in accordance to the gloss conventions. While the test reference sentence is more strict and solely presents a sentence using und and oder. The predicted sentence is more in the context of the conversation about deafness.

Original Gloss Input	MAMA1A* PAPA8* $GEST^ PAPA1B* OMA1C*
Model Predicted Sentence	Meine Mutter und mein Bruder, der ja auch hörend war.
Test Reference Sentence	Ja, MAMA und PAPA, oder so PAPA, PAPA, OPA, OMA.

Figure 9
Showing the BLEU-scores from the 4 performed sub-experiments. The description of the experiments can be found in section 5.1. Overall there is a trend of improvement along with the fine-tuning of the model. On the X-axis the different models are presented. On the Y-axis the score is shown, the calculation and description of the BLEU score is described in Section 4.5)



6.2 Experiments

The second experiment was performed to answer the question of whether or not adding temporal data into the source files would aid in the performance of the model. The translations generated by the temporal model received a BLEU-score of 2.85 and a TER of 0.97. When compared to the baseline system this a decrease of 22.8% in BLEU-score and a similar TER score, an example of a generated translation can be seen in (Table 4).

The third experiment was performed to answer the question of whether vocal data would aid in the performance of the baseline model. The results show a BLEU-score of 2.03 which is a decrease of 45.0% compared to the baseline. The calculated Translation Error Rate on the model is 1, the maximum rate since TER-scores are between 0 and 1, suggesting that a high number of edits was necessary to reach the reference. A practical example of this can be seen in (Table 5). Research, however, suggests vocal utterances of deaf individuals are a strong indication of meaning (Konrad et al., 2020).

The final experiment was performed to answer the question of whether or not combining the third and fourth experiment would aid in the performance of a gloss to

Table 4
Showing the translation results of the baseline model, the original input is the input of the glosses in accordance with the gloss conventions with embedded temporal features denoted by |NUMBER. The model seems to break down while translating, repeating the same word "Mama" over and over again.

Original Gloss Input	MAMA1A* | 120 PAPA8* | 340 $GEST^ | 60 PAPA1B* | 260 OMA1C* | 120
Model Predicted Sentence	Das ist meine Mama und meine Mama und meine Mama meine Oma
Test Reference Sentence	Ja, MAMA und PAPA, oder so PAPA, PAPA, OPA, OMA.

Table 5
Showing the translation results of the vocal model, the original input is the input of the glosses in accordance with the gloss conventions with embedded vocal features denoted by |mouthing. The sentences of the model seem to be quite a bit more basic as compared to the baseline and temporal experiments.

Original Gloss Input	GUT1* | nan TOLL1A | toll GLÜCK1* | glück GUT1* | nan
Model Predicted Sentence	Das war sehr gut.
Test Reference Sentence	Gut, das ist toll. Du hattest Glück.

text machine translation system. Results show a BLEU-score of 1.69, which is a decrease of 54.2% compared to the baseline. The calculated TER on the model is 0.97, while a small increase compared to the baseline. Surprisingly, however, while being worse compared to the baseline the combined TER was on par with the temporal experiment and better compared to the vocal experiment. An example of the translation can be found in (Table 6).

Table 6
Showing the translation results of the combined experiment, the original input is the input of the glosses in accordance with the gloss conventions with embedded vocal temporal features denoted by |mouthing|NUMBER. As is shown the sentence does not resemble the reference sentence anymore.

Original Gloss Input	GUT1* | nan | 100 TOLL1A | toll | 440 GLÜCK1* | glück | 160 GUT1* | nan | 160
Model Predicted Sentence	Das ist wirklich gefährlich..
Test Reference Sentence	Gut, das ist toll. Du hattest Glück.

Figure 10
Showing the results from the 4 performed experiments. On the X-axis the different models are presented. On the Y-axis the score is shown, the calculation and description of the BLEU score is described in Section 4.5)

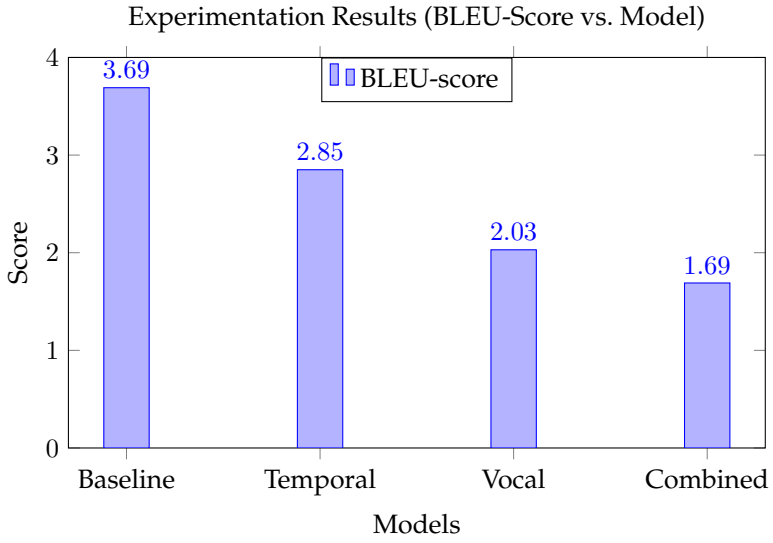
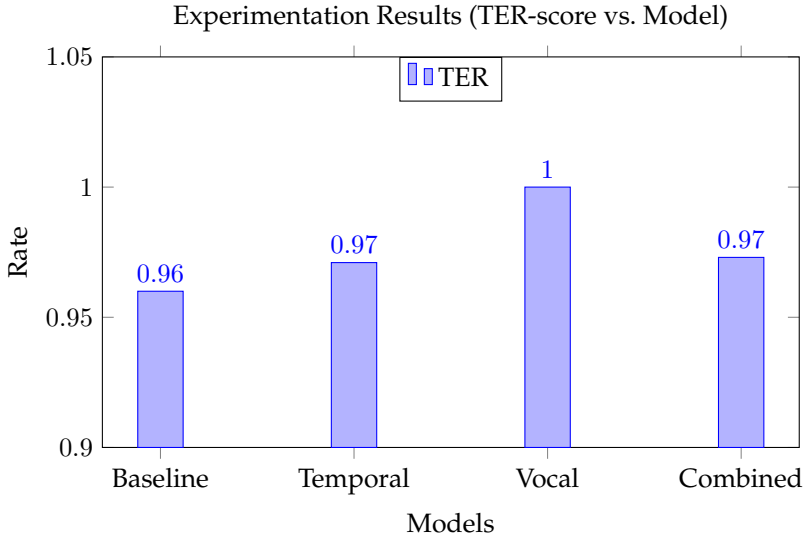


Figure 11
Showing the results from the 4 performed experiments. The range of the TER is [0, 1] while 0 is without any edits and 1 with a lot. On the X-axis the different models are presented. On the Y-axis the score is shown, the calculation and description of the Translation Error Rate is described in Section 4.6



7. Discussion

The goal of this thesis was to research to what extent temporal and vocal data aids in the performance of a MT system for sign glosses to text. The hypothesis stated that it was to be expected, based on prior research (Konrad et al., 2020), for temporal and vocal embeddings to increase the performance of the system.

7.1 Research findings

NMT systems were created to aid with translating sign glosses to text using temporal features. It was shown that the addition of temporal and vocal features to the baseline did not aid in increasing the BLEU-score of the model. In fact, the addition of temporal and vocal data decreased the baseline BLEU score by as much as 50%. The results suggest that neither temporal nor vocal data aids in the performance of a machine translation system for sign language glosses to text.

7.2 Results

The hypothesis stated that the use of token data (be it temporal or vocal) would help increase the accuracy of the baseline model. The results show the contrary, a non-aided simple gloss to text translation system outperformed the aided temporal and vocal systems. It was hypothesised that vocal utterances would in fact aid more in the performance of a machine translation system. However, the baseline system outperforms the vocal system by as much as 45%. The vocal experiment has the worst TER rate, especially considering the maximum rate is 1. The reason for this drop in performance is likely due to two aspects:

1. **The lack of data** Because of the small size of the corpus, only barely enough data was used to train the system, in combination with small hardware capabilities compared to contemporary research heavily decreased the expected accuracy. This was due to both time constrains and hyperparameters optimization. Since, the hyperparameters were not tuned with ideal circumstances in mind but resource management. As was suggested in Qi et al. (2018), word features are the most efficient in models trained on small datasets. However, if the dataset is at the bare minimum, it shows a significant drop-off in efficiency results. The lower performance of the temporal, vocal, and combined experiments supports this claim.
2. **Increase in vocabulary, decrease in commonality** Because of the non-standardization of the tokens (i.e.: |40, |200 instead of |time_0-200) the vocabulary increased exponentially, giving the model not many common words to train on and increasing the amount of OOV-words. A major flaw in the research was choosing the wrong dataset. Since the DGS Corpus (Konrad et al., 2020a) was created for purposes of conservation, it over-represented local dialects. This resulted in the model upon translation receiving more unknown words. This is caused by, for example, the sign Kuh (Cow) being different between Nordrhein-Westfalen and Sachsen.

Another reason might be because of the vocal and combined data files being pre-processed differently, due to the several different encoding errors the BPE algorithm

malfunctioned for these file. Therefore, it was decided to not utilize the algorithm for these systems. This is clearly shown in the results, however, the average increase in BLEU score after the use of the algorithm was shown to be around 40%. Even if the BPE algorithm was applied on the Vocal and Combined in the ideal circumstance would increase their respective scores to 2.8 and 2.5. This could be solved by creating a library that has the feature for BPE embedded into its code, like MarianNMT (Junczys-Dowmunt et al., 2018). While MarianNMT was considered for the purposes of this Thesis, due to the restrictions applied by Google Colab and its incompatibility with Microsoft Windows 10, the use of MarianNMT was discontinued. Preliminary results showed a substantially lower BLEU-score compared to the OpenNMT-pipeline, presumably caused by its lack of running time and insufficient hyperparameters.

7.3 Alternatives and Future Research

The results suggest that neither temporal nor vocal data aid in the performance of a machine translation system for sign language glosses to text. Supporting the results found by Camgoz et al. 2018; Qi et al. 2018. The research, therefore, sits among an increasingly growing SLT literature suggesting that embedding words into an NMT system decreases the accuracy rate of the model.

Future research should be conducted into different forms of pre-processing the tokens. By the grouping of tokens into small subgroups, there might be a reduction in over-diversity in the vocabulary, whereas in this paper the words could be paired with temporal features spanning the range from 0 to 1000, having 5 distinct groups could possibly have an effect on the performance of the system.

A suggested alternative baseline is performed by the stripping of the gloss-conventions (Konrad et al., 2020) from the data files, as was practised by Othman & Jemni (2012). Preliminary results show that an improvement of 11.3% compared to the current baseline, however, that would undermine the concept of sign language and return to SLR style translation by not taking into account linguistical aspects (Camgoz et al., 2018). Therefore, future research should focus on the creation and standardisation of an annotated German Sign Language corpus using less diverse tokens while maintaining proper grammar by proper sentence creation.

8. Conclusion

The field of Signal Language Recognition has shown promising results regarding sign to text translation. However, it fails to take into account the linguistical aspects of sign language. Therefore systems were needed with the ability to encode the linguistical aspects. It was established that neural machine translation systems were most suitable for this task. The goal of this study was to integrate temporal data into neural machine translation systems and research its effectiveness. This was accomplished by using factored neural machine translation with source word features. We conducted four experiments. The baseline showed the ability to translate rudimentary gloss sentences into similar spoken German sentences. It was found that by adding temporal data the accuracy of the system decreased while increasing the number of errors. Even though contemporary research suggests that vocal utterances correlate with the meaning of the sign, it was found that inserting vocal data did not increase the accuracy of the model. Combining the latter two experiments into one model resulted in a model that saw a 50% decrease in accuracy rate. This implies assigning temporal, vocal, or vocal-temporal data directly to the words translated by a gloss-to-text translation

system does not improve its effectiveness. Future research could be done by training a full spatiotemporal gesture to spoken language system and embedding both the word and additional features separately.

The Self-Reflection can be found in the Appendix.

References

Admasu, Y. F. & Raimond, K. (2010). Ethiopian sign language recognition using artificial neural network. In *2010 10th International Conference on Intelligent Systems Design and Applications*, (pp. 995–1000).

Akmeliawati, R., Ooi, M. P.-L., & Kuang, Y. C. (2007). Real-time malaysian sign language translation using colour segmentation and neural network. In *2007 IEEE Instrumentation Measurement Technology Conference IMTC 2007*, (pp. 1–6).

Aouiti, N., Jemni, M., & Semreen, S. (2015). Arab gloss annotation system for arabic sign language. In *2015 5th International Conference on Information & Communication Technology and Accessibility (ICTA)*, (pp. 1–6). IEEE.

Barone, A. V. M., Haddow, B., Germann, U., & Sennrich, R. (2017). Regularization techniques for fine-tuning in neural machine translation. *arXiv preprint arXiv:1707.09920*.

Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Leveling, J., Monz, C., Pecina, P., Post, M., Saint-Amand, H., et al. (2014). Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, (pp. 12–58).

Braem, P. B. & Brentari, D. (2001). Functions of the mouthing component in the signing of deaf early and late learners of swiss german sign language,". *Foreign Vocabulary in Sign Languages: A Cross-Linguistic Investigation of Word Formation, D. Brentari, Ed. Mahwah, NJ: Erlbaum*, 1–47.

Bragg, D., Koller, O., Bellard, M., Berke, L., Boudreault, P., Braffort, A., Caselli, N., Huenerfauth, M., Kacorri, H., Verhoef, T., et al. (2019). Sign language recognition, generation, and translation: An interdisciplinary perspective. In *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*, (pp. 16–31).

Brugman, H., Russel, A., & Nijmegen, X. (2004). Annotating multi-media/multi-modal resources with elan. In *LREC*. Citeseer.

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).

Camgoz, N. C., Koller, O., Hadfield, S., & Bowden, R. (2020). Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 10023–10033).

Camgoz, N. C., Saunders, B., Rochette, G., Giovanelli, M., Inches, G., Nachtrab-Ribback, R., & Bowden, R. (2021). Content4all open research sign language translation datasets. *arXiv preprint arXiv:2105.02351*.

Chan, A. (2012). Overview of bleu.

Cooper, H., Holt, B., & Bowden, R. (2011). Sign language recognition. In *Visual analysis of humans* (pp. 539–562). Springer.

Crasborn, O. & Sloetjes, H. (2008). Enhanced elan functionality for sign language corpora. In *6th International Conference on Language Resources and Evaluation (LREC 2008)/3rd Workshop on the Representation and Processing of Sign Languages: Construction and Exploitation of Sign Language Corpora*, (pp. 39–43).

De Coster, M., Van Herreweghe, M., & Dambre, J. (2020). Sign language recognition with transformer networks. In *12th International Conference on Language Resources and Evaluation*.

Dias, D. B., Madeo, R. C., Rocha, T., Biscaro, H. H., & Peres, S. M. (2009). Hand movement recognition for brazilian sign language: a study using distance-based neural networks. In *2009 international joint conference on neural networks*, (pp. 697–704). IEEE.

Fels, S. S. & Hinton, G. E. (1993). Glove-talk: A neural network interface between a data-glove and a speech synthesizer. *IEEE transactions on Neural Networks*, *4*(1), 2–8.

Gage, P. (1994). A new algorithm for data compression. *C Users Journal*, *12*(2), 23–38.

García-Martínez, M., Barrault, L., & Bougares, F. (2016). Factored neural machine translation architectures. In *International Workshop on Spoken Language Translation (IWSLT'16)*.

Glenn, C. M., Mandloi, D., Sarella, K., & Lonon, M. (2005). An image processing technique for the translation of asl finger-spelling to digital audio or text. In *Instructional Technology and Education of the deaf Symposium, Rochester, NY*, (pp. 1–7).

Hanke, T., Hong, S.-E., König, S., Konrad, R., Langer, G., Matthes, S., Nishio, R., & Regen, A. (2019). Segmentation.

Hanke, T., König, S., Konrad, R., Langer, G., & Rathmann, C. (2009). Dgs corpus project - development of a corpus based electronic dictionary german sign language /german.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Holden, E.-J., Lee, G., & Owens, R. (2005). Australian sign language recognition. *Machine Vision and Applications*, *16*(5), 312–320.

Huang, J., Zhou, W., Li, H., & Li, W. (2015). Sign language recognition using 3d convolutional neural networks. In *2015 IEEE international conference on multimedia and expo (ICME)*, (pp. 1–6). IEEE.

IDGS (2020). Background information. - dgs-korpus. https://www.sign-lang.uni-hamburg.de/meinedgs/ling/start_en.html.

Johnston, T. et al. (2008). Corpus linguistics and signed languages: no lemmata, no corpus. In *3rd Workshop on the Representation and Processing of Sign Languages*.

Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., Seide, F., Germann, U., Fikri Aji, A., Bogoychev, N., Martins, A. F. T., & Birch, A. (2018). Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, Melbourne, Australia.

Kellett, C. J. M. & Ochse, E. (2008). English in international deaf communication.

Kiani Sarkaleh, A., Poorahangaryan, F., Zanj, B., & Karami, A. (2009). A neural network based system for persian sign language recognition. In *2009 IEEE International Conference on Signal and Image Processing Applications*, (pp. 145–149).

Klein, G., Kim, Y., Deng, Y., Senellart, J., & Rush, A. (2017). OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, (pp. 67–72)., Vancouver, Canada. Association for Computational Linguistics.

Koller, O., Forster, J., & Ney, H. (2015). Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer Vision and Image Understanding*, *141*, 108–125.

Konrad, R., Hanke, T., Langer, G., Blanck, D., Bleicken, J., Hofmann, I., Jeziorski, O., König, L., König, S., Nishio, R., Regen, A., Salden, U., Wagner, S., Worseck, S., Böse, O., Jahn, E., & Schulder, M. (2020a). Meine dgs – annotiert. öffentliches korpus der deutschen gebärdensprache, 3. release / my dgs – annotated. public corpus of german sign language, 3rd release. https://doi.org/10.25592/dgs.corpus-3.0.

Konrad, R., Hanke, T., Langer, G., Blanck, D., Bleicken, J., Hofmann, I., Jeziorski, O., König, L., König, S., Nishio, R., Regen, A., Salden, U., Wagner, S., Worseck, S., Böse, O., Jahn, E., & Schulder, M. (2020b). Meine dgs – annotiert. öffentliches korpus der deutschen gebärdensprache, 3. release / my dgs – annotated. public corpus of german sign language, 3rd release. https://www.sign-lang.uni-hamburg.de/meinedgs/html/1413451-11105600-11163240_de.html.

Konrad, R., Hankse, T., Langer, G., König, S., König, L., Nishio, R., & Regen, A. (2020). Public dgs corpus annotation conventions.

Kristoffersen, J. H., Troelsgård, T., Langer, G., Hanke, T., Konrad, R., & König, S. (2016). Designing a lexical database for a combined use of corpus annotation and dictionary editing. In *7th Worshop on the Representation and Processing of Sign Languages: Corpus Mining*.

Kshirsagar, S. (2002). A multilayer personality model. In *Proceedings of the 2nd international symposium on Smart graphics*, (pp. 107–115).

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

Liang, R.-H. & Ouhyoung, M. (1996). A sign language recognition system using hidden markov model and context sensitive search. In *Proceedings of the ACM symposium on virtual reality software and technology*, (pp. 59–66).

Maraqa, M. & Abu-Zaiter, R. (2008). Recognition of arabic sign language (arsl) using recurrent neural networks. In *2008 First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT)*, (pp. 478–481). IEEE.

Meadow, K. P., Greenberg, M. T., Erting, C., & Carmichael, H. (1981). Interactions of deaf mothers and deaf preschool children: Comparisons with three other groups of deaf and hearing dyads. *American Annals of the Deaf*, 454–468.

Mittal, A., Kumar, P., Roy, P. P., Balasubramanian, R., & Chaudhuri, B. B. (2019). A modified lstm model for continuous sign language recognition using leap motion. *IEEE Sensors Journal*, *19*(16), 7056–7063.

Murakami, K. & Taguchi, H. (1991). Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, (pp. 237–242).

Nijmegen: Max Planck Institute for Psycholinguistics, T. L. A. (2020). Elan (version 6.0) [computer software]. https://archive.mpi.nl/tla/elan.

Oomen, M. (2017). Iconicity in argument structure: Psych-verbs in sign language of the netherlands. *Sign Language & Linguistics*, *20*(1), 55–108.
OpenNMT (2018). How do i use the transformer model? https://opennmt.net/OpenNMT-py/FAQ.htmlhow-do-i-use-the-transformer-model.
Othman, A. & Jemni, M. (2012). English-asl gloss parallel corpus 2012: Aslg-pc12. In *5th Workshop on the Representation and Processing of Sign Languages: Interactions between Corpus and Lexicon LREC*.
Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, (pp. 311–318).
Paradis, J., Nicoladis, E., & Genesee, F. (2000). Early emergence of structural constraints on code-mixing: Evidence from french–english bilingual children. *Bilingualism: Language and cognition*, *3*(3), 245–261.
Parton, B. S. (2006). Sign language recognition and translation: A multidisciplined approach from the field of artificial intelligence. *Journal of deaf studies and deaf education*, *11*(1), 94–101.
Perniss, P. M. (2007a). *Space and iconicity in German sign language (DGS)*. PhD thesis, Radboud University Nijmegen Nijmegen.
Perniss, P. M. (2007b). *Space and iconicity in German sign language (DGS) Section 1.1*. PhD thesis, Radboud University Nijmegen Nijmegen.
Pfaff, C. W. (1979). Constraints on language mixing: Intrasentential code-switching and borrowing in spanish/english. *Language*, 291–318.
Pietrandrea, P. (2002). Iconicity and arbitrariness in italian sign language. *Sign Language Studies*, 296–321.
Porta, J., López-Colino, F., Tejedor, J., & Colás, J. (2014). A rule-based translation from written spanish to spanish sign language glosses. *Computer Speech & Language*, *28*(3), 788–811.
Prillwitz, S., Hanke, T., König, S., Konrad, R., Langer, G., & Schwarz, A. (2008). Dgs corpus project–development of a corpus based electronic dictionary german sign language/german. In *3rd Workshop on the Representation and Processing of Sign Languages: Construction and Exploitation of Sign Language Corpora*, (pp. 159).
Qi, Y., Sachan, D. S., Felix, M., Padmanabhan, S. J., & Neubig, G. (2018). When and why are pre-trained word embeddings useful for neural machine translation? *arXiv preprint arXiv:1804.06323*.
Sedley, D. (2003). *Plato's Cratylus*. Cambridge University Press.
Sennrich, R., Haddow, B., & Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.
Shapira, D. & Storer, J. A. (2002). Edit distance with move operations. In *Annual Symposium on Combinatorial Pattern Matching*, (pp. 85–98). Springer.
Shterionov, D. (2020). Python: train_test_dev.py.
Sloetjes, H. (2017). Elan. In *The Oxford handbook of corpus phonology*.
Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200. Citeseer.
Starner, T. & Pentland, A. (1997). Real-time american sign language recognition from video using hidden markov models. In *Motion-based recognition* (pp. 227–243). Springer.
Starner, T., Weaver, J., & Pentland, A. (1998). Real-time american sign language recognition using desk and wearable computer based video. *IEEE Transactions on pattern analysis and machine intelligence*, *20*(12), 1371–1375.
Stokoe Jr, W. C. (2005). Sign language structure: An outline of the visual communication systems of the american deaf. *Journal of deaf studies and deaf education*, *10*(1), 3–37.
Thissen, G. (2021). Multiple feature gloss to text translation. https://github.com/GijsThissen/g-thissen-csai-thesis.
Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, (pp. 4489–4497).
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.
Vogler, C. & Metaxas, D. (1999). Parallel hidden markov models for american sign language recognition. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, (pp. 116–122). IEEE.

1. Self-Reflection

At first, when I started this thesis, I didn't quite know what I was getting myself into. The careful planning it would involve, the research, the literature review it was all kind of new to me. While I experienced many of these concepts in the earlier parts of my bachelor combining them proved to be a difficult task. What proved especially difficult was the planning, since I am used to being able to do an assignment or learn for an exam within the span of a week, however, I learned that regarding a research project this is not the case. Especially since my topic is Deep Learning related it took careful planning and prediction on how long each particular experiment would take and whether it would fit within the time span I had. There were some foreseeable setbacks, mainly the introduction of CAPTCHA in Google Colab, that caused the experiments to take longer than expected which took me by surprise. Especially since I had to start from scratch again. What I learned from this was that my planning (dis)ability is the biggest bottleneck in my research. If I had planned out the experiments in a better way by doing a prototype beforehand these problems could have been avoided. There should have been a better literature review on my part, mainly regarding the available architectures, instead of struggling with the dataset using the Phoenix dataset as described in (Camgoz et al., 2018) would have made for a better thesis. What I learned from this experience is that when it comes to research planning, prototyping and exploring is essential, but for me at first surprising, part of the project. When it came to programming I had some prior experience regarding Neural Machine Translation, however, having prior experience does not equate to a smooth ride. Mainly due to not exploring the available libraries well enough. I did not realize soon enough that the library I ended up using OpenNMT (Klein et al., 2017) also supported word embedded features, therefore I decided upon MarianNMT (Junczys-Dowmunt et al., 2018) a library that is both incompatible with my system and had to be run externally. Throughout the thesis, I did start to learn not only how to use the libraries but also the inner workings since I frequently had to adjust certain functions that were not suitable for my experiments. I often ran into trouble regarding certain parts of my code that would have easily been resolved if I had a variable explorer enabled. Furthermore, due to the thesis, I had my first experience with reading a paper and applying its code instead of finding it on GitHub or there being a library present, something that was an insightful experience and will undoubtedly be useful in the future. As well as writing a proper README file in my GitHub repository (Thissen, 2021) that gave me insight into how users would view my code. Before the start of this project, my main focus when it came to NMT was related to Dutch to English translation, however, when I started this thesis project I was immersed into the world of the deaf. Using sign language to communicate is really resilient, adapting to the situation you are in, and therefore deserves more scientific research done on the subject. Especially when it wasn't considered a language until fairly recently (Stokoe Jr, 2005). Having said that I thought it was quite hard to explain my thoughts on the subject especially in the Introduction and Related Works section as I was overwhelmed by all the information that came to me. Looking back I should have listened more to my supervisor when he said that we had to create a file with all the summaries of the papers that we read. Instead of doing that I had a bunch of different files on my PC, outside my PC, and inside my head that caused a giant mess. While this approach is considered acceptable when it comes to small essays, with a huge thesis this becomes an impossible task. What I learned from this is that I need to be more organised. While I do not have a problem with speaking in public, I do have trouble with speaking rehearsed text in public. Due to the COVID-19 restrictions present, it made it more difficult since I

couldn't look the listeners in the face making me stutter quite a bit. However, I learned from this experience that I have to know my text inside-out and I need to rehearse more.

Appendix B: Merging Code

```
class create_file(object):
    def __init__(self, path="datasets/"):
        self.path = path
        self.inputs = listdir(str(path))
        self.dataframe = self.drop_empty()

    def clean_dataframe(self, dataset):
        """
        Load the dataset into a frame, delete the "Unnamed" column,
        and replace all instances of nothing with a numpy nan.
        Requires the numpy library.

        :param dataset: str, input name of the dataset.csv
        :return: dataframe, a cleaned dataframe with only used columns
        """
        dataframe = pd.read_csv(self.path + str(dataset), sep=",")  # Read into
     a frame
        dataframe = dataframe.loc[:, ~dataframe.columns.str.contains("^Unnamed"
    )]  # Drop the "^Unnamed" column
        dataframe.replace("", np.nan, inplace=True)  # Replace the empty values
     with a nan

        return dataframe

    def merge(self):
        """
        The EAF-files consist of different persons, this functions merges those
        into one dataframe.

        :return: Dataframe. A merged dataframe consisting of all users.
        """
        combined = pd.DataFrame()
        for dataset in self.inputs:
            temp = self.clean_dataframe(dataset)

            # Normalize the column names into the translated versions.
            column_list = list(temp.columns)
            normalized_names = ["Time", "Right", "Mouth", "Translation", "Left"
    ]
            # A dictionary is created with the corresponding column_list name
    and the normalized name
            translation_dict = {column_list[n]: normalized_names[n] for n in
    range(len(normalized_names))}
            temp = temp.rename(columns=translation_dict)

            # Combine the dataframes into one universal dataframe
            combined = combined.append(temp, ignore_index=True, sort=False)

        return combined

    def list_definer(self, input_list):
        """
        Finds the True instances in a list and stores their indexes.

        :param input_list: list, a list of True's and False's.
        :return: list, the list of indexes that were true in the input_list.
        """
        output_list = []
        # Looping over a enumerated input_list
```

```
        for number, element in enumerate(input_list):
            # If the element is True append the index else continue the loop
            if element:
                output_list.append(number)
            continue
        return output_list

    def drop_empty(self):
        """
        Dropping the empty rows from the dataset causing it to become more information packed.
        Downsides of this approach can be found in the Discussion of the written Thesis.

        :return: dataframe, a dataframe where there are no empty rows.
        """
        combined = self.merge()

        # Find the empty rows for each respective token (time excluded since it is always present).
        left_sign = set(self.list_definer(list(combined['Left'].isnull().values)))
        right_sign = set(self.list_definer(list(combined['Right'].isnull().values)))
        mouth_token = set(self.list_definer(list(combined['Mouth'].isnull().values)))

        # Find the intersection of these tokens.
        signs = left_sign.intersection(right_sign)
        empty_rows = signs.intersection(mouth_token)
        signs_missing = list(empty_rows)

        # Dropping the empty rows
        final_dataframe = combined.drop(signs_missing)

        return final_dataframe
```

Listing 1
Shown is the Python code to merge the multiple CSV-files into one DataFrame. (Thissen, 2021)

Appendix C: Fixing

```
def fixing():
    """
    Fixes files by removing the extra \n that was created during the
    train_test_dev.py splitting of the data. OpenNMT
    does not handle empty lines well and will assign "translations".
    """
    # Get current working directory
    working_directory = str(os.getcwd())

    # Calculate the total amount of files in the working directory
    dataset = os.listdir(str(working_directory))
    total = len(dataset)

    print("\nDeleting white spaces in files\n")

    # Source (regarding the lines of code related to the tqdm-libary):
    # DDGG. (2018, Feb 22) tqdm not showing bar. Stackoverflow.com.
    # https://stackoverflow.com/questions/48935907/tqdm-not-showing-bar

    with tqdm(total = total) as pbar:
        for element in dataset:
            # Create new name
            new_name = "f-" + element
            with open(str(element), "r", encoding="utf-8") as f:  # Read from
    this file
                with open(str(new_name), "w+", encoding="utf-8") as fixed:  #
    Write in this file
                    # Removing all the extra white spaces
                    while True:
                        line = f.readline()
                        if line == "":
                            break
                        if line == "nan\n":
                            continue
                        if not line.isspace() and line != "nan":
                            fixed.write(line)
            pbar.update(1) # Update pbar by 1
```

Listing 2
Shown is the Python code to fix the false white lines created by the train_test_dev.py script. (Thissen, 2021)

Appendix D: BPE Modification

```
            ##############
            # Normal
            ##############
            for item in new_word[:-1]:
                output.append(item + self.separator)
            output.append(new_word[-1])


            ###############
            # EDIT 11/05/2021 Gijs Thissen
            # This is to "fix" the problems I am having with DGS glosses with
    features that get split up
            # Run code above for the normal way
            ###############
            # If new_word contains more than one element (having one element
    suggest there not being a temporal feature)
```

```
            if len(new_word)>1 and new_word[-1][-1].isdigit(): # Last element
    of last element of new_word is digit
                for item in new_word[:-1]:
                    output.append(item + self.separator + new_word[-1])
            else:
                for item in new_word[:-1]:
                    output.append(item + self.separator)
                output.append(new_word[-1])
```

Listing 3
Shown is the Python code that is a modification to the normal Byte Pair Encoding algorithm. (Thissen, 2021)