Text-to-Filipino Sign Language Translation

Sergei Allen C. Bugayong, John Christopher D. Carnate, Francis Xavier L. Guimalan,
Dyson Sigmond B. Malto, Reggie C. Gustilo
Gokongwei College of Engineering
De La Salle University, Manila, Philippines
sergei_allen_bugayong@dlsu.edu.ph, john_christopher_carnate@dlsu.edu.ph, francis_guimalan@dlsu.edu.ph,
dyson_sigmond_malto@dlsu.edu.ph, reggie.gustilo@dlsu.edu.ph

***Abstract*—Approximately 430 million individuals globally experience hearing impairment or speech deprivation, presenting challenges in social interaction for the Deaf community. Deaf individuals often encounter difficulties in engaging with others, leading to feelings of isolation, depression, and distress. Despite the widespread availability of interpreters, achieving seamless conversational exchanges remains a persistent challenge. Furthermore, in the Philippines, American Sign Language is prevalent, especially in rural areas, rather than the native Filipino Sign Language. This research paper introduces a communication system designed to translate text into Filipino Sign Language Utilizing diverse algorithms such as Long Short-Term Memory (LSTM) for text buffering and lookup tables, the researchers conducted thorough evaluations in collaboration with interpreters and Deaf individuals. The outcomes of these assessments revealed a commendable level of reliability, with the system consistently achieving an accuracy rate surpassing 90%. Participants, assessed through a Likert scale, expressed a high degree of satisfaction with various aspects of the system, including its design, relevance, user-friendliness, and effectiveness.**

***Index Terms*— Deaf, machine learning, Filipino sign language, Tagalog text , LSTM, Encoder-Decoder, Natural Language Processing, Python**

I. INTRODUCTION

A. Study Background

In human social interaction, effective communication plays a pivotal role in the exchange of information among individuals (Dam, 2017). The necessity for engaging with diverse individuals arises from pursuing common goals and fostering mutual understanding within society. Sign language, a non-verbal communication system primarily employed through manual gestures, body language, and facial expressions, serves as a means of communication for individuals with hearing and speech impairments (Pfau et al., 2012). Facilitating dialogue sign language extends a fundamental human right to those facing challenges in hearing and speaking (Sign et al. to Deaf People's Rights, 2018).

In the Philippines, awareness of the existence of Filipino Sign Language remains limited among the populace despite widespread recognition of American Sign Language (Mendoza, 2018). Mendoza contends that 70% of the Filipino Deaf community, predominantly situated in urban areas, employs FSL, which has developed in relative isolation from mainstream linguistic influences. The Department of Education's (DepEd) K–12 curriculum underscores the auditory-vocal nature of spoken languages, such as Filipino, in contrast to the visual-spatial characteristics of sign languages like FSL, which adhere to distinct linguistic structures encompassing phonology, morphology, syntax, and discourse (Guno, 2019). Despite these linguistic attributes, barriers persist, impeding effective communication for Deaf individuals attempting to convey their thoughts to the general populace (Xu et al., 2021).

People grappling with hearing loss and speech impediments encounter challenges in communication due to a scarcity of interpreters worldwide, resulting from high demand (Vasquez, 2019). Technological advancements offer a promising solution to this predicament, as exemplified by Aasfowa et al.'s (2018) project introducing a speech-to-sign language gesture software model. This innovative approach translates spoken language into sign language gestures, presented through a 3D animated avatar, thereby enhancing accessibility and inclusivity for individuals with hearing and speech impairments.

B. *Problem Statement*

According to World Health Organization (WHO) statistics, more than 5% of the global population experiences disabling hearing loss (McPhillips, 2022). Despite the utilization of sign language, individuals facing challenges in hearing and speech encounter communication difficulties, particularly when the availability of interpreters is limited globally due to rising demand (Vasquez, 2019). Typically, interpreters are present when a Deaf person needs to communicate, especially in large gatherings where communication involves addressing numerous individuals (Young, 2019). However, in the absence of interpreters, these individuals tend to withdraw from society, feeling disconnected from those with average hearing abilities. The need for interpreters often results in hearing and speech-impaired individuals being excluded from communication exchanges. In a typical dialogue, individuals with normal hearing take turns using auditory cues; when a Deaf individual attempts to participate in a conversation, they frequently encounter difficulties in navigating an ongoing dialogue (Meek, 2020). With this context in mind, this project aims to develop a text-to-sign language system.

C. *Objective*

The objective is to develop a text-to-sign language system incorporating a look-up table and a machine learning-driven Tagalog text correction model. This system aims to facilitate the conversion of textual content into Filipino Sign Language (FSL). The specific goals of this study are delineated as follows:

- Employing a machine learning algorithm to effectively translate letters, words, and interactive sentences into FSL.
- Implementing a look-up table and text as a point of reference and buffer to enhance the efficiency of translation operations.
- Attaining a reliability range of no less than 90% accuracy in the translation process.
- Subjecting the system to practical testing with a Deaf individual to evaluate the confidence level in communication, assessed through qualitative feedback obtained from respondents in a survey format.

D. *Scope and Delimitations*

The following are the project's scope and delimitations:

1. The system is designed to convert the Filipino alphabet, encompassing letters A-Z (excluding ñ and ng), numbers 0-10, 50 words, and 30 commonly used phrases, facilitating bidirectional translation between text and sign language.
2. For sign language-to-text operations, the device will present output on a monitor/LED screen, whereas the text-to-sign language functionality will generate images/videos displayed through a slideshow.
3. The system is not configured to acquire new vocabulary, numerals, or phrases during its operational state.
4. In instances where a specific word or phrase is unrecognized, the system will systematically translate it on a letter-by-letter basis.

E. *Related Studies*

In Machine Learning, Natural Language Processing (NLP) is a critical conduit between human and computer language. Specifically applied in this project, NLP is utilized for Grammar Error Correction within the English language. The investigation involves a comparative analysis of various models employing distinct algorithms. The dataset comprises 500,000 data points extracted from two publicly accessible language repositories. The models under consideration incorporate Encoder-Decoders, LSTM algorithms, Attention Mechanisms, and Inference Algorithms. The evaluation metric employed is the BLEU Score, with the deployed model achieving a score of 0.5469 and an Inference Time of 164 seconds. The model demonstrates effectiveness and is recommended for application across diverse datasets (Gandhi, 2021).

Additionally, a proposed speech-to-sign communication model is introduced to facilitate communication between individuals with hearing impairments and those without (Aasfowa et al., 2018). The model is intended for use by the Deaf community in Gujarat, incorporating a specific sign language tailored to the characteristics of the local Deaf community. The proposed model's operational workflow entails converting spoken language from a non-deaf individual into text, followed by transforming this text into HamNoSys notation. HamNoSys serves as a phonetic transcription system universally applicable to all sign languages, featuring a direct correspondence between symbols and gestures, including hand location, placement, movement, and shapes. Subsequently, HamNoSys notations undergo conversion into SiGML codes, which, in turn, serve as inputs for an avatar animation

displaying the Gujarat sign language to the Deaf individual. The research underscores the one-way communicative efficacy of the proposed model, enabling seamless communication from non-deaf individuals to the Deaf population without the need for intermediary devices (transistors) (Aasfowa et al., 2018).

II. THEORETICAL CONSIDERATIONS

A. Natural Language Processing (NLP)

Natural Language Processing or known as NLP is a field in machine learning that makes computers understand any natural language by translating that language into data that the computer can understand. This field involves many processes and data manipulation. NLP based algorithms can include tokenization, tagging, sentiment analysis algorithms. Meanwhile, NLP based applications is very wide and not limited to translation tasks. Currently, NLP applications are used by search engines such as Chat-GPT and Bing AI, to make access to data easier to society (Bird et al., 2009).

B. Recurrent Neural Network (RNN)

Recurrent Neural Network or known as RNN is a type of Neural Network that is made specifically for sequential data. Traditional Neural Network such as the Feed-Forward has the disadvantage of not having a memory variable that allows them to remember past inputs. RNN can retain this information by having a hidden state variable called (h_t). This makes the RNN very suitable to applications where the order of the inputs will significantly affect the output (Nabi, 2021).

1) Long-Short Term Memory (LSTM)

Traditional RNNs have problems with long inputs due to large number of dependencies. To handle this problem, researchers inserted a memory cell with gating mechanisms into the model architecture. In this way, the model will be able to select what to retain and forget at each step. This model architecture is called Long-Short Term Memory or LSTM.

A LSTM model architecture consist of a memory cell that can store the dependencies while having to do many time steps. This memory cell has gating mechanisms that is from three different gates. These gates are the input, forget, and output gates. An LSTM model can instruct itself to forget or retain any dependency depending on the current input at each time step (Srivastava, 2017).

III. DESIGN CONSIDERATION

A. Sequence-to-Sequence Architecture (Seq2Seq)

Sequence-to-Sequence architectures or Seq2Seq were made to solve machine problems that involves transforming a specific sequenced input into a corresponding output. The model is not limited have the input and output sequence to be the same (Muñoz, 2020). Due to this, this model is very useful in translation tasks. In this study, the proponents decided to use this architecture to translate possible wrong Tagalog input into the corrected version. One of the key architectures of Seq2Seq is the Encoder-Decoder Model. This model involves three main components. the encoder model, the decoder model, and the hidden state vector (Jurafsky & Martin, 2021). A simple Encoder-Decoder model is shown in Figure 1.

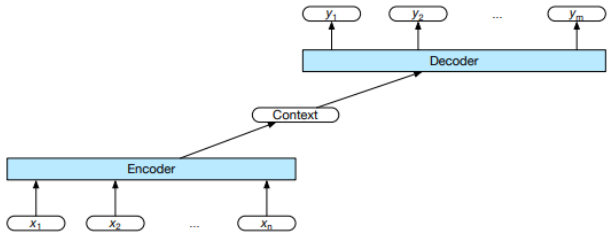

Figure 1. Encoder-Decoder Model

B. Attention Mechanism

Attention Mechanism is made to make Natural Language Processing Models understand and get the context based on a long input. This algorithm works by creating and utilizing attention weights. In this algorithm, the machine initially calculates an attention score to predict the output. In this study, this attention score is calculated using dot product. This score would be the basis for the attention weights. This weight instructs the model to what elements the model should prioritize. In this way, the model can understand the context of a long and complex input (Erdem, 2022).

C. Beam Search Algorithm

Search Algorithm is an algorithm that instructs a program on how to search and access this specific information. This algorithm is recommended to be used by NLP models so that the model can balance the memory resources and the prediction quality. There are many types of Search Algorithms. In this study, the proponents used the Beam Search Algorithm. Beam Search maintains a set of multiple

candidate sequences at each step. This would make the model more flexible and avoid being stuck at one candidate. In exchange for this, this is more computationally expensive than other simpler search algorithms (Topper, 2023).

D. Bilingual Evaluation Understudy (BLEU)

Bilingual Evaluation Understudy is a metric for a generated sentence from an input sentence. This is done by comparing the n-grams from the output sentence from the input or reference sentence (Brownlee, 2019). The scoring is from 0 to 1. A score of zero means that the generated sentence is a complete mismatch to the input sentence. While having a score of 1 means that the generated sentence is a complete match. Generally, a score 0.6 to 0.7 indicates a high quality of translation while avoiding a high chance of overfitting. This score metric is available as a function in the Python Natural Language Toolkit Library (NLTK).

E. TensorFlow

Implementing a Deep Learning model can be done in different ways or framework. TensorFlow is an open-source framework developed by Google. It supports different machine learning applications such as Natural Language Processing (NLP) and Computer Visions (Tensorflow, 2023). This framework or library has several built-in functions that makes it easier for any Machine Learning Engineer to design and implement a machine learning model. There are other alternatives other than TensorFlow, one of the is PyCharm. However, the proponents choose to use this framework due to more available functions (Doshi, 2023).

IV. METHODOLOGY

This section illustrates and explains a detailed creation of the Text to FSL program. The program flowchart is shown in Figure 2. The goal of the program is to translate 26 letters, numbers 0-10, 50 basic words, and 30 common phrases into its corresponding FSL gestures. As shown in the flowchart, the program contains a grammar and spelling Tagalog corrector using Deep Learning. The generated output of the model will be sent to the Text to Picture algorithm to create the gestures using lookup tables and a buffer. Lastly, in case the FSL output is wrong, there is a feedback algorithm that would correct the system.

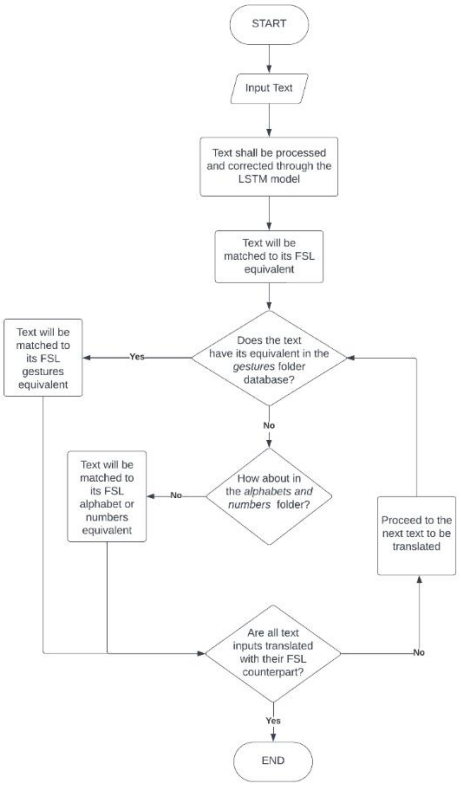

Figure 2. Text to FSL Program Flowchart

A. Development of Model Dataset

1) Raw Datapoints

The raw datapoints that was used for the model has two columns. The first column is any possible input that a user might type while using the program. This can be either correct or wrong Tagalog sentences, phrase, or word. The second column is the corrected Tagalog sentences, phrase, or word. A wrong first column is defined if there is a spelling or grammar mistake in the Tagalog text. If the first column is already correct, then the second column has no changes. Figure 3 shows an example of the raw dataset.

	A	B
53	sa agosto at disyembre ako aalis	sa agosto at disyembre ako aalis
54	oo kuya	oo kuya
55	naiintindihan ko sinabi mo	naiintinhindihan ko sinabi mo
56	magandangg araw sa lahat	magandang araw sa lahat
57	sa guwebes at sabadtro	sa huwebes at sabado
58	patensya na	pasensya na
59	mabuhau ka	mabuhay ka
60	kamusta kayooo	kamusta kayo
61	maagandang umaaagaa	magandang umaga
62	magandang umaga nak	magandang umaga anak
63	indi na nila ako sinasabay	hindi na nila ako sinasabay
64	kain tayo sa tetyembre	kain tayo sa setyembre
65	walang pasok sa enero	walang pasok sa enero
66	sagado na naman	sabado na naman
67	mas mahal pa mga bilihin ngayon kesa noor	mas mahal pa mga bilihin ngayon kesa
68	nalungkot sila sa nangyari	nalungkot sila sa nangyari
69	jusq ayan na nilagnat na ako	jusko ayan na nilagnat na ako
70	grabe ang pagnanakaw ng pera ng bayan	grabe ang pagnanakaw ng pera ng bay
71	nagpapatuloy ang kada oras na mga misa	nagpapatuloy ang kada oras na mga m
72	di pede magusap	hindi pwede magusap
73	umalis kanaaaaaa	umalis ka
74	minsan lang ako bumati	minsan lang ako bumati
75	sa totoo lang hindi ko rin alam	sa totoo lang hindi ko rin alam
76	ayaw ko sinisipon	ayaw ko sinisipon
77	inilarawan ng pulisya ang babae na nasa bel	inilarawan ng pulisya ang babae na na
78	kaya pala nilagnat ako kahapon	kaya pala nilagnat ako kahapon

unprocessed_tagalog

Figure 3. Unprocessed Dataset

The whole dataset was from three sources. The first main source was from the researchers. These datapoints includes the correct and wrong version of the 116 targeted Tagalog sentences, phrase, or word. Other datapoints are from the language website, Tateoba. This website has different downloadable text files that contains a single language (Tateoba, n.d). Lastly, another source is from a Tagalog Corpuz by Cruz et.al (2019). This Tagalog Corpuz contains long Tagalog Sentences that can be used to train the model for longer inputs. In total, the number of datapoints is 115, 357.

2) Preprocessing

To lessen the computation burden and make the data uniform, there proponents decided to preprocess the raw dataset. Table 1 and Table 2 show the changes set by the proponents in this stage. The proponents also decided to make all characters in lowercase and erased duplicated inputs. This part of the preprocessing was done using an automation algorithm and used the regular expression or regex library. Regex library is a tool that allows a user to find a specific characters or symbols in a database (Wanyoike, 2023). This would allow them to easily manipulate these characters or symbols even in a large dataset. Figure 4 illustrates the cleaned dataset. The third column indicates that the datapoint has been cleaned.

	A	B	C
1	processed_input	processed_output	y
2	sya ay mganda at masunurin	siya ay maganda at masunurin	1
3	kamusta ka naman	kamusta ka naman	1
4	magandang hapon po	magandang hapon po	1
5	di ko rin alam bat nagkaganyan	hindi ko rin alam bakit nagkaganyan	1
6	sa buwan ng nobyembre ako magsisimula	sa buwan ng nobyembre ako magsisimula	1
7	ang hirap taalga ng trabaho na ito	ang hirap talaga ng talaga ng trabaho na ito	1
8	malungkot sila ate at kuya sa nangyari	malungkot sila ate at kuya sa nangyari	1
9	ang puting ibon	ang puting ibon	1
10	ngayon ay kaarawan ng kaapatid q	ngayon ay kaarawan ng kapatid ko	1
11	at dahil dun naging masaya araw nya	at dahil dun naging masaya araw niya	1
12	hindi mo inaabot sa oras	hindi mo inaabot sa oras	1
13	nakabatay sa kanyang lokasyon	nakabatay sa kanyang lokasyon	1
14	isang dambuhalang tau	isang dambuhalang tao	1
15	aak ilang taon ka	anak ilang taon ka	1
16	lunes martes miyerkules kuya	lunes martes miyerkules kuya	1
17	kain tayo bukas	kain tayo bukas	1
18	mahal kamusta ka	mahal kamusta ka	1
19	ipinanganak ako sa lunes	ipinanganak ako sa lunes	1
20	san ka nakatirah	saan ka nakatira	1
21	magandang gabi sa inyo	magandang gabi sa inyo	1
22	sa hapon kami nagusap	sa hapon kami nagusap	1
23	nagmahalan c tatay at nnay	nagmahalan si tatay at nanay	1
24	masigla sya nagising	masigla siya nagising	1
25	pangalan ko ay	pangalan ko ay	1
26	ang pangalan nya ay	ang pangalan niya ay	1

Figure 4. Processed Dataset

Table 1. List of Erased Special Characters

Special Characters
:
;
-
!
/
\
?
,
_
(
)
Additional White Space

Table 2. List of Corrected Tagalog Words or Phrases

Shortened Word/Phrase	Corrected Version
Anong	ano ang
ako'y	ako ay
tayo'y	tayo ay
't	At
sa'yo	sa iyo
ba'ng	ba ang
na'to	na ito
'to	Ito
sila'y	sila ay
mo'y	mo ay
'wag	Huwag
samu't	samu at
nuna'y	nuna ay
nila'y	nila ay
'yo	Iyo
nati'y	natin ay
ngayo'y	ngayon ay
ko'y	ko ay

After this, the 115, 357 raw datapoints were divided into three datasets. These are the training, validation, and testing datasets. The division is also done using automation, the automation is also instructed to split the datapoints in a random order. After the division, the number of processed datapoints for the Training, Validation, and Testing datasets are 91362, 22842, and 1151 respectively. In total, the used datapoints are 115, 355. This leaves 2 raw datapoints unused. Figures 5, 6, and 7 show a screenshot of the Training, Validation, and Testing dataset respectively.

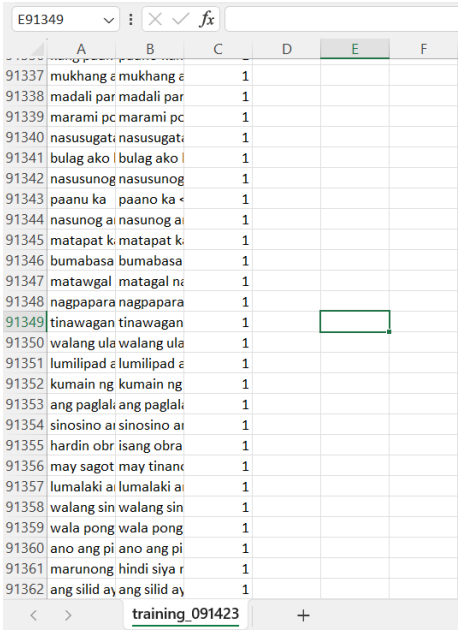

Figure 5. Training Dataset

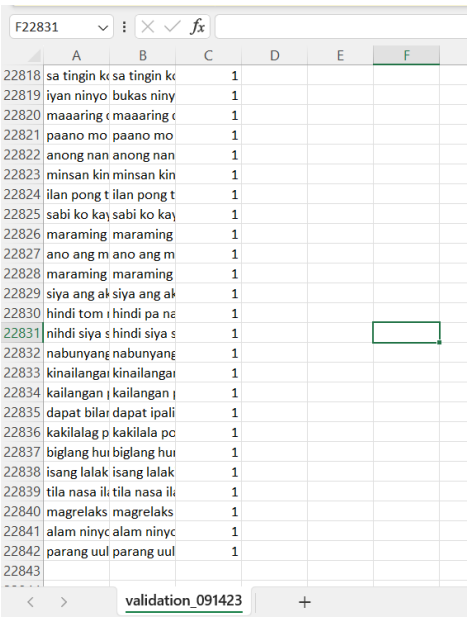

Figure 6. Validation Dataset.

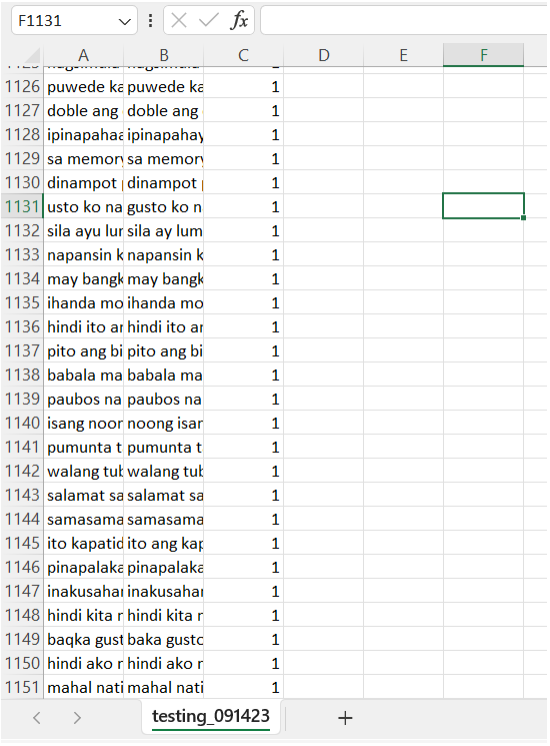

Figure 7. Testing Dataset

Before it was trained into the model, these datapoints need to be tokenized. Tokenization is a preprocess where the text data is broken into small units, called tokens, for the machine to get specific

dependency at each token. The tokenization of the datapoints is adding a start and end tokens at each datapoints. The tokenization consists of two parts, the first tokenization is for the input tokenizer. The input tokenizer is the tokenization for the first column or the possible encoder inputs. In total, there is a total of 39, 927 unique tokens found. The second part is for the output tokenizers. This is the tokenization for the second column or the decoder outputs. For this tokenizer, there is a total of 23, 434 unique tokens found.

B. Development of the Text-to-Text Model

To make the model deployable in its application, the model needs to have a fast inference time but at the same time a high quality of translation. The model architecture used in this study is an Encoder-Decoder Model that uses Dot Product and Beam Search for its Attention and Search Algorithms respectively. The model was implemented using the TensorFlow Framework. According to a comparison study by Gandhi (2021), this model architecture is working at a balanced quality based on its performance and computation balance. The first part of the model architecture is the data pipeline. The data pipeline prepares the data to be feed to the model.

The next part of the model architecture are the classes for the encoder-decoder model. These classes contain a part of the model. These classes are the Encoder, Decoder, Attention Mechanism. After this point, the model is trained with early stopping mechanism for 50 epochs. The last part of the model is the search algorithm which would help the model decode the output and the BLEU score in order to evaluate the translation quality.

C. Development of the Lookup Table and Buffer

The lookup table consists of the 116 targeted Filipino words, numbers (0 to 9), letters (a to z, without the inclusion of ñ and ng), and phrases. This is listed on Table 3 as shown below. The lookup table of the program is divided into two folders. The first folder is the lookup table for the FSL, in webp format, Filipino words and phrases. Meanwhile, the second folder consists of the FSL, in gif format, of the Filipino letters and numbers. A screenshot for both folders are illustrated in Figure 8 and 9 respectively.

TABLE 3.
DATASET OF ALPHABET, NUMBERS, WORDS, AND PHRASES

Category	Model	Inclusions
Alphabet & Numbers	Alphabet and Numbers	A
		B
		C
		D
		E
		F or Siyam
		G
		H
		I
		K
		L
		M
		N
		O or Sero
		P
		Q
		R
		S
		T
		U
		V or Dalawa
		W or Anim
		X
		Y
		Isa
		Tatlo
		Apat
		Lima
		Pito
		Walo
	J & Z (FTA)	J
		Z
Pagbati	Pagbati – 1 (FTA)	Pasensya
		Pagbati
		Paalam
		Magandang Umaga
		Magandang Hapon
	Pagbati – 2 (FTA)	Magandang Gabi
		Nagagalak akong makita ka
		Mahal kita
		Ingat
		Ang pangalan ko ay
Lugar	Lugar (FTA)	Bahay
		Taas
		Ilalim
		Simbahan
		Doon

		Dito
Tao	Tao -1 (FTA)	Sanggol
		Kaibigan
		Nanay
		Ate
	Tao – 2 (FTA)	Tatay
		Kuya
		Pamilya
Panahon	Panahon	Ulap
		Maliwana
		Madilim
		Ulan
		Araw
Panghalip	Panghalip – 1 (FTA)	Sila
		Saakin
		Saatin
		Sayo
	Panghalip – 2 (FTA)	Ikaw
		Ako
		Tayo
Sagot	Sagot – 1 (FTA)	Oo
		Hindi
		Baka
		Teka
		Salamat
	Sagot – 2 (FTA)	Walang Anuman
		Makikiraan
		Naiinitindihan ko
		Hindi ko naiintindihan
Oras	Oras (FTA)	Mamaya
		Linggo
		Bukas
		Kahapon
		Oras
		Gabi
		Umaga
		Ngayon
Basic FSL	Basic FSL (FTA)	Masama
		Dahil
		Problema
		Ulit
		Masaya
		Malungkot
		Ayos
Pandiwa	Pandiwa (FTA)	Dasal tayo
		Alam ko yan
		Tingin Dito
		Ihinto ang sasakyan
		Tara, Kain tayo
		Aral lang ako
Tanong	Tanong – 1 (FTA)	Kamusta Ka
		Saan bahay mo
		Ilang taon ka na
		Saan ang paaralan mo
		Ano ang pangalan mo
	Tanong – 2 (FTA)	Anong araw ngayon
		Alin doon
		May sakit ka ba
		Ano ang sinabi mo
		Kailan tayo magkikita
	Tanong -3 (FTA)	Ano ang ginagawa mo
		Kailan ka pinanganak
		Pwede 8 akita matulungan
		Magkano po ito
		Paano mo sabihin ang

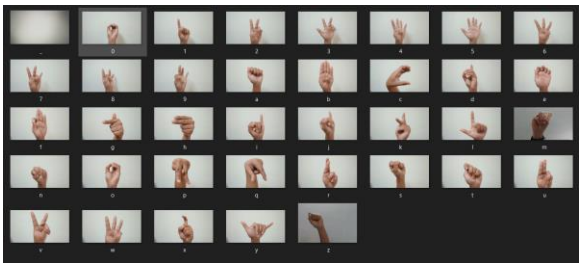

Figure 8. Alphabets and Numbers Dataset Folder

Figure 9. Gestures Dataset Folder

Considering the buffer's role as a provisional data storage space, the application can utilize the Text-

to-Text model and the output directory within the Text-to-Formal Sign Language (FSL) directory. This configuration establishes a framework for managing input and its corresponding output, facilitating the retention of previously translated text results. The archival repository of past translation outputs is a valuable resource for data accumulation, potentially streamlining subsequent data handling. It is pertinent to highlight that the directory also preserves processed. GIF files generated from antecedent textual inputs allow independent examination without necessitating program execution.

D. Development of the Text-to-Picture Program

The Text-to-Picture program mainly consists of the conversion from the generated text output of the Spelling and Grammar Error Correction Model into its equivalent FSL gestures. The algorithm was done by splitting the generated text into letters. These letters were store in an array and the computer is instructed to check if there is a match. However, in this way, there would be a problem with a white space. The algorithm would match the text into an unintended file/s. For example, if the generated output is “kailan “, the program would match it to the FSL file “kailan ka pinanganak”. To solve this, the proponents changed the naming convention of the FSL files and the generated output. In this case, the proponents made all whitespace into an underscore, “_”. This special character does not affect the program since the generated text was cleaned and preprocess by the previous algorithm.

The next feature of the Text-to-Picture Program is the feedback system. The feedback system was created to correct the limitations of the model. Due to limitations of the database and computing resources, the model is limited on what it is trained to. This system is based on a reference text file named “user_feedback.txt”. This text file contains the original input that was typed by the user and the expected output of the user. The user can type the expected output in the text file and the next time the same input is typed, the program will prioritize the expected output in the text file, rather than going through the process again.

E. Testing Procedure

The testing procedure of the program was mainly divided into two parts. The first part was the evaluation metrics for the spelling and grammar error correction model. This includes the number of actual epochs, losses, and the BLEU scores. The next part of testing was its accuracy in real time usage and acceptability. The testing was done with FSL interpreters and deaf people. This part of the testing was divided into two. The first is the accuracy test of the program. The FSL interpreters were asked to test the accuracy of the system by typing the whole 116 targeted texts and texts that is not belong to the targeted 116. The interpreters were also encouraged to intentionally type an incorrect input to test the correction model. For the FSL deaf participants, they were asked to test a portion of the system. The 116 targeted text were divided into sets and the participants were asked to choose and tests one set.

The next part of the testing is the acceptability of the Text to FSL program. The proponents used a Likert-scale containing a score from 0 (the lowest) to 10 (the highest). The form contains different questions that asks their confidence rating for the different aspects of the program. Some of these include ease of usability, accuracy, and gesture quality. The main goal of the testing is to attain a score of at least 90% accuracy and know the confidence level from the deaf community.

V. RESULTS AND DISCUSSION

A. Model Metrics

For the evaluation metrics of the model, the actual number of the training is 48 epochs before it went to early stopping to avoid overfitting. Using an AMD Ryzen 7 5800H processor, the training period is estimated to be around 23 hours. A screenshot of the model parameters is shown in Figure 8. As shown in the figure, the BLEU score is approximately to be around 0.66. This shows a high quality of translation while having a low chance of overfitting. Overfitting is a machine learning problem where the model fails to generalize the training data. In this condition, the model will memorize the training data and would perform poorly on unseen or new inputs (Baheti, 2023).

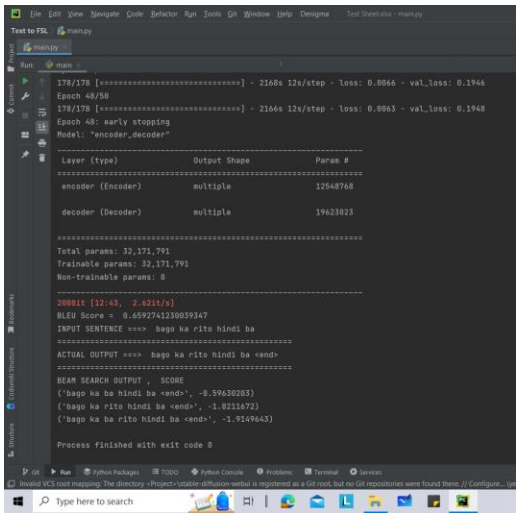

Figure 10. BLEU Score

Another model metric is the accuracy losses of the model. The figure below illustrates the losses of the model during the training process. It is observed in the graph that the training and validation losses gradually decrease as the number of epoch increases. A gradual decrease in training loss means that the model is performing good generalization of the dataset. Meanwhile, a manual decrease in the validation loss means that the model is performing good prediction on unseen data (Baeldung, 2023).

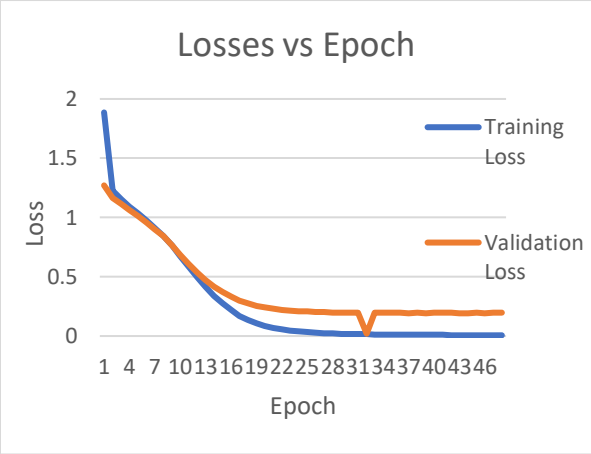

Figure 11: Training Loss Graph

B. Testing Accuracy

As mentioned, the accuracy testing of the Text-to-FSL program is divided into two parts. The first part is the testing for the FSL interpreters. In total, there are five (5) interpreters that tested the program. The results of each testing are illustrated by Figure X and the testing summary by Figure 10. The highest accuracy that the program got is 96.03. Meanwhile, the lowest is at 95.69. In total, the average accuracy is approximately to be 96%. This means that 96% of the 116 targeted texts were translated into FSL gestures properly.

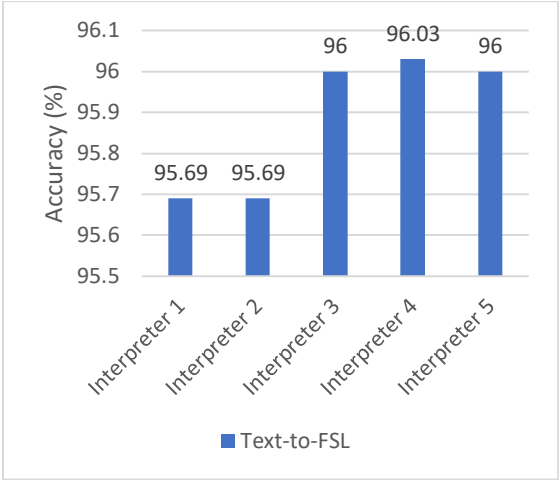

Figure 12. Accuracy Scores for the Interpreters

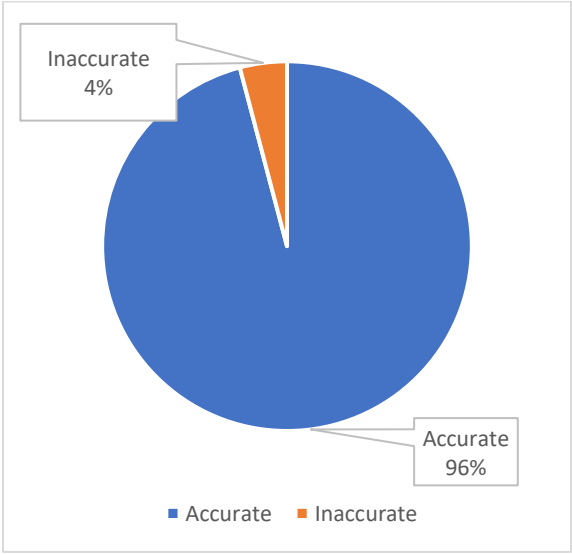

Figure 12. Summary of the Accuracy Scores for the Interpreters

The next part is the accuracy testing for the deaf. In this study, there was a total of 15 participants that tested the program. The deaf did not test the whole 116 targeted words but was chose a set to test. Figure 14 shows the accuracy testing of each participant. As illustrated, the highest accuracy that was achieved was 100%. It means that all the tested inputs were accurate and understandable by the participants. Meanwhile, the lowest accuracy score was at 88.89%. Overall, the

testing accuracy for deaf is 96%. Both accuracy scores for the interpreters and deaf exceeded the objective of 90%.

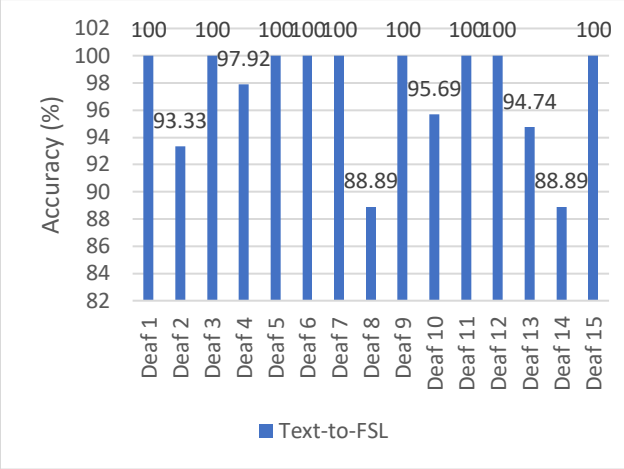

Figure 14. Deaf Participants Accuracy Scores

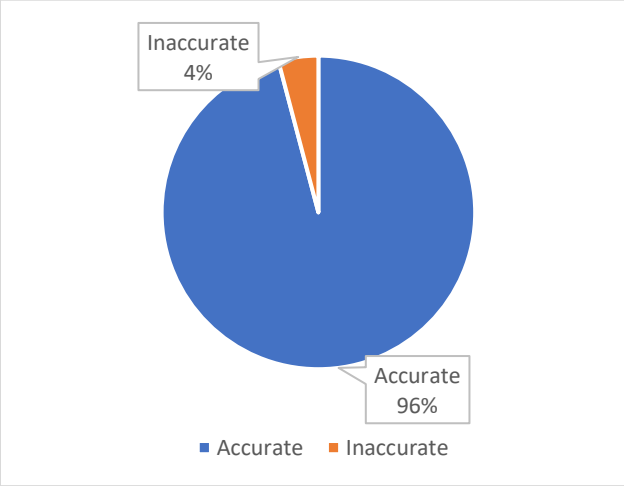

Figure 15. Summary of Deaf Participants Accuracy Scores

C. *Qualitative Metrics*

The deaf participants answered a form that has asks the confidence score of the specific set that these participants chose with. The form is a Likert scale that has a scaling of 0, the lowest score, to 10, the highest possible score. Table 4 shows the metrics for each participant in their chosen sets. As illustrated by the table below, the lowest confidence level was at 3 for the Tao set. Meanwhile, there were three participants that rated Basic FSL set the highest score of 10.

Table 4. Text-to-FSL Scores per Categories

Categories	Text-to-FSL Scores									
	1	2	3	4	5	6	7	8	9	10
Alphabet and Numbers				1				2		5
Pagbati								1		2
Lugar									1	1
Tao			1							3
Panahon								1	2	3
Panghalip							2			1
Sagot					1					3
Oras				1						1
Pandiwa								1		
Tanong								1	1	4
Basic FSL								2	3	1
TOTAL	**0**	**0**	**1**	**2**	**1**	**0**	**2**	**8**	**7**	**24**

VI. CONCLUSION

Overall, the objectives of the Text-to-FSL program were met. First, the proponents were able to create a working and successful program that converts an input text into its FSL gesture using Machine Learning. Second, the accuracy of the program was tested and has a total accuracy score of 96% for both FSL interpreters and deaf participants. This was more than the actual goal of 90%. Lastly, deaf community answered and gave their confidence levels through a quantitative survey.

Even though it is a successful study, the proponents have some areas and aspects of the program for future and interested researchers to work

upon. First recommendation is the comparison and utilization of other NLP model architectures such as transformers. Transformers are model architectures that does not rely on attention mechanisms <Source>. The second recommendation is the optimization and improvement of the deep learning model. Future researchers can improve the model through optimization techniques such as fine-tuning and make a larger dataset for the model. Lastly, the researchers recommend adding more languages and sign languages to the program. This research may extend to have a cross language feature such as Tagalog to Arabic Sign Language.

The program has several potential directives that it may be use from. First and foremost, it can be developed into an offline application. This project would involve software development and a larger database for the model to train upon. There is also a possibility to change the input into speech. This would create a speech to FSL application. Lastly, this program can be integrated to communication applications such as messenger.

VII. REFERENCES

Baeldung. (2023). Training and validation loss in deep learning | Baeldung on Computer Science. *Baeldung on Computer Science*. https://www.baeldung.com/cs/training-validation-loss-deep-learning

Baheti, P. (2023, April 24). What is Overfitting in Deep Learning [+10 Ways to Avoid It]. *V7*. https://www.v7labs.com/blog/overfitting

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*.

Brownlee, J. (2019). A gentle introduction to calculating the BLEU score for text in Python. *MachineLearningMastery.com*. https://machinelearningmastery.com/calculate-bleu-score-for-text-python/

Cruz, J. C. B., Tan, J. A., & Cheng, C. (2020). Localization of Fake News Detection via Multitask Transfer Learning. European Language Resources Association (ELRA).

Doshi, P. (2023, July 17). *TensorFlow vs. PyTorch: Which Deep Learning Framework is Right for You?* Stackify. https://stackify.com/tensorflow-vs-pytorch-which-deep-learning-framework-is-right-for-you/

Erdem, K. (2022, January 6). Interactive guide to Attention Mechanism by Kemal Erdem (burnpiro) | May, 2021 | Towards Data Science. *Medium*. https://towardsdatascience.com/introduction-to-attention-mechanism-8d044442a29

Gandhi, P. (2022, January 5). Grammar Error Correction with Deep Learning - Towards Data Science. *Medium*. https://towardsdatascience.com/grammar-error-correction-af365dad794

Jurafsky, D., & Martin, J. (2021). *Encoder-Decoder Models, Attention, and Contextual Embedding* (2nd ed.). https://web.stanford.edu/~jurafsky/slp3/old_dec21/10.pdf

Muñoz, E. (2020, October 12). A guide to the Encoder-Decoder model and the attention mechanism. *Medium*. https://betterprogramming.pub/a-guide-on-the-encoder-decoder-model-and-the-attention-mechanism-401c836e2cdb

Nabi, J. (2021, December 10). Recurrent Neural Networks (RNNs) - towards data science. *Medium*. https://towardsdatascience.com/recurrent-neural-networks-rnns-3f06d7653a85

Srivastava, P. (2017, December 10). Essentials of Deep Learning : Introduction to Long Short Term Memory. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/

Tatoeba. (n.d.). What is tatoeba?. https://tatoeba.org/en/about

Topper, N. (2023). Introduction to the Beam search Algorithm. Built In. https://builtin.com/software-engineering-perspectives/beam-search

Vashee, K. (2022). Understanding MT quality: BLEU scores. ModernMT Blog. https://blog.modernmt.com/understanding-mt-quality-bleu-scores/

Wanyoike, M. (2023, October 5). Learn Regex: A Beginner's Guide. SitePoint. https://www.sitepoint.com/learn-regex/