AI Quantification of Language Puzzle to Language Learning Generalization

A Project Presented To

The Faculty of Department of Computer Science

San Jose State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

By

Harita Shroff

May 2020

The Designated Project Committee Approves the Master's Project Titled

AI Quantification of Language Puzzle to Language Learning Generalization

By

Harita Shroff

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2020

Dr. Christopher Pollett	Department of Computer Science
Dr. Katerina Potika	Department of Computer Science
Kevin Smith	Department of Computer Science

ACKNOWLEDGEMENT

Foremost, I would like to express my deep sense of gratitude to my advisor Dr. Christopher Pollett for the continuous support of my project, for his patience, motivation, enthusiasm and immense knowledge. I could not have imagined having a better advisor and mentor for this project.

Besides my advisor, I would like to thank the rest of my project committee: Dr. Katerina Potika and Prof. Kevin Smith, for their encouragement, guidance and valuable feedback.

Finally, I thank my wonderful family for their countless hours of support and encouragement along the way.

ABSTRACT

Online language learning applications provide users multiple ways/games to learn a new language. Some of the ways include rearranging words in the foreign language sentences, filling in the blanks, providing flashcards, and many more. Primarily this research focused on quantifying the effectiveness of these games in learning a new language. Secondarily my goal for this project was to measure the effectiveness of exercises for transfer learning in machine translation. Currently, very little research has been done in this field except for the research conducted by the online platforms to provide assurance to their users [12]. Machine learning has been used in this research to achieve the goals mentioned earlier. Specifically, deep learning models with Recurrent Neural Network (RNN) were employed to process the data. Models were designed on popular exercises from these platforms using sequence-to-sequence learning. Our research discovered that most of the models had cross-validation accuracy in the range of 70%-80%. This result shows that knowledge learned from one model is transferrable to the other.

Index terms: **Recurrent Neural Network (RNN), Deep Learning, Sequence-to-sequence Learning, Machine Translation.**

Table of Contents

1 INTRODUCTION 7

2 BACKGROUND 12

2.1 NEURAL NETWORK 12

2.2 RECURRENT NEURAL NETWORK 16

2.3 WORD EMBEDDINGS 18

2.4 SEQUENCE-TO-SEQUENCE MAPPING 18

3 DESIGN AND IMPLEMENTATION 22

3.1 ENVIRONMENT 22

3.2 DATASET 23

3.3 ACTIVTIES AND DATASET PROCESSING 25

3.4 MODEL ARCHITECTURE 31

4 EXPERIMENTS AND RESULT 35

4.1 MODELS 35

4.2 ACCURACY 36

4.3 TRAINING 37

4.4 TESTING 38

5 CONCLUSION AND FUTURE WORK 41

6 REFERENCES 43

1 INTRODUCTION

Learning a foreign language is always a challenging task. The spread of the internet not only connected different parts of the world but also provided some powerful tools. Online language learning is one such tool people find appealing nowadays. There are many tools available on the internet which can help a user learn a new language at their own pace. According to Efficacy of New Language Application by R. Vesselinov and J. Grego, very little research has been done to study the effectiveness of these platforms [12]. These platforms use a different set of activities including language puzzles to help its users learn a new language. According to Wing Yee, signs are always meaningful, and all forms of meaning-making should be taken seriously [5]. Different modes of expressing signs are writing, speech, and images. This multimodal approach in online language learning platforms imitates different activities. Figure 1 shows sample exercises from the Duolingo platform. Other online platforms use a similar format of exercises for users to learn new languages. The goal of this research was to quantify the effectiveness of these methods using artificial intelligence technology. As these platforms have become popular, the question this research was trying to answer is the effectiveness of these exercises. The knowledge a user acquires from different exercises is transferable across them or not was the focus. Artificial intelligence is used to measure the effectiveness of these platforms. One of the key techniques for this project was machine learning language models. According to Eugene Charniak, instruments that want to translate sentences from one language to the other need to have the capability to differentiate between sentences [2]. A language model is the formalization of this idea. As mentioned by Saini and Sahula [8], translating a sentence from one language to the other is easy if the structure of the languages is similar. In a case where structures are different, and sentences are taken as a sequence, sequence-to-sequence technique can be applied for the translation. A sequence-to-sequence neural network model takes a sequence of words

from one language as an input and produces a sequence of words in another language. This term will be defined further in details in section 2.4. The approach we employed here was to have different sequence-to-sequence models for different activities shown in figure 1.

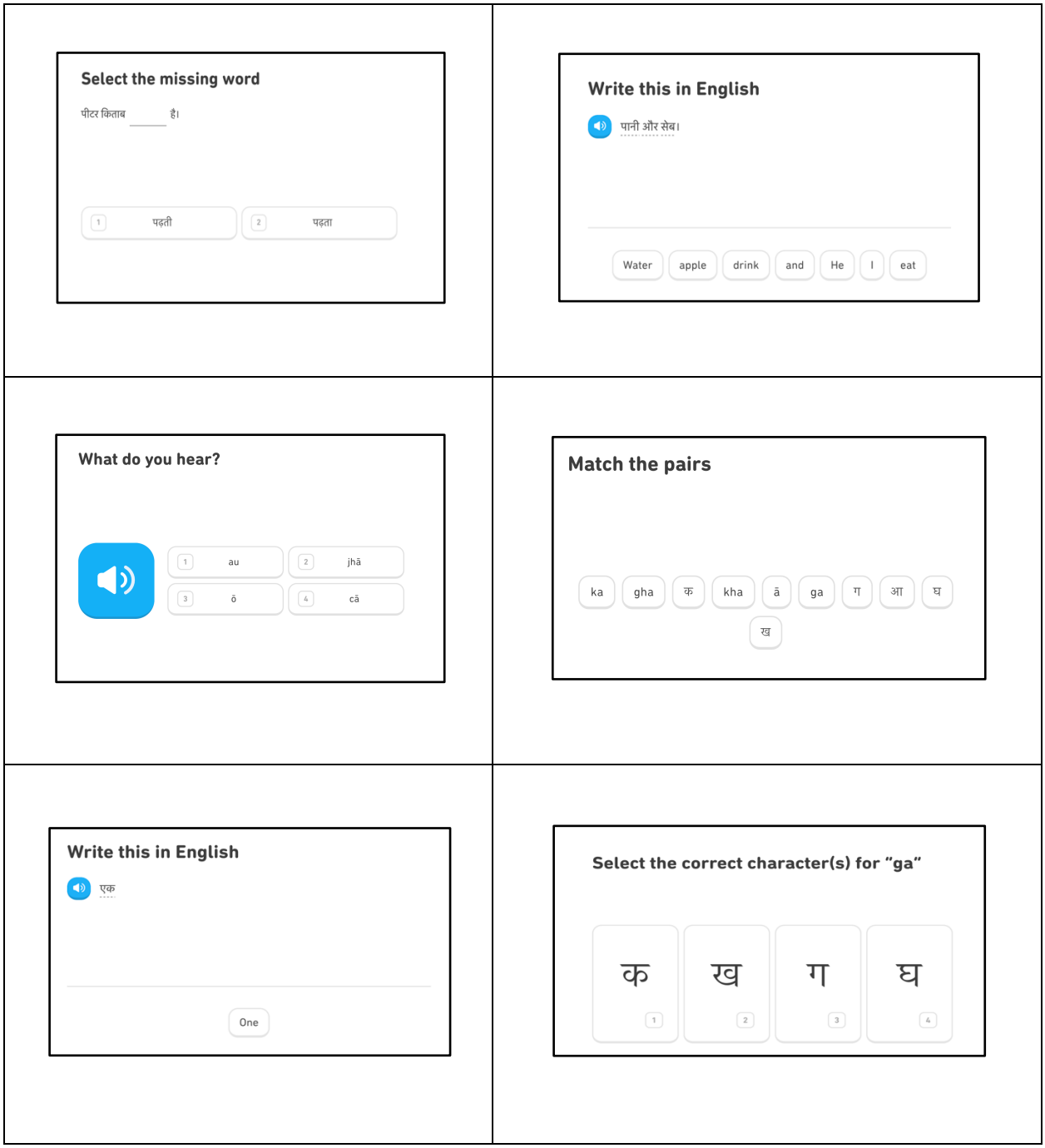

Figure 1. Different Duolingo activities to learn Hindi language.

The aim of a user going through these exercises is to learn a new foreign language. The idea in this project is going to be aligned to that goal. We will design our neural network models to learn the Portuguese language with the help of different exercises. Hence the model definition was decided by the activity type. For example, if the activity is fill-in-the-blanks, the model can map a set of the sequence where source language sentence is mixed up with translated sentence sans a word to a target sequence containing the exact translation. Similarly, for making pair and writing sentence activities model can be made to map a sequence of source sentences along with translated sentences and some extra random words from the target language to a sequence of exact translation.

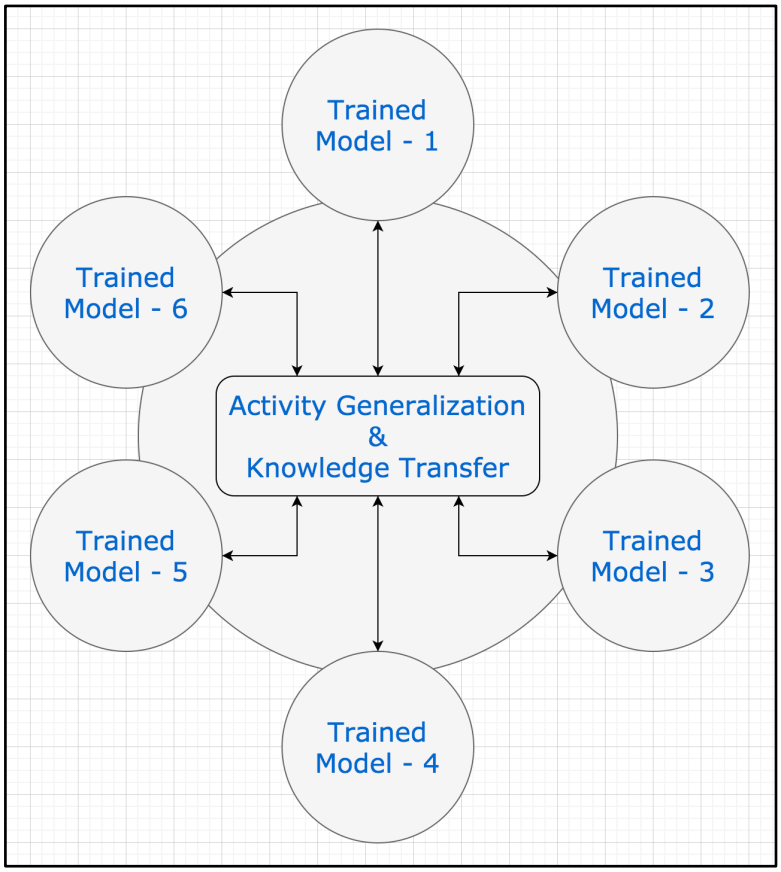

Figure 2. High level design used for the project.

Figure 2 shows the high-level design for this project. Different models were trained on the same set of data. The trained models were saved for other models to test. Other models used their test data to check the accuracy of the data. Dataset was selected such that the data can be modeled into mimicking different activities. Simple English to Portuguese language translations were selected from Duolingo as well as the Tatoeba project. These sentences were divided into train and test sets. Each model had its train and test data converted based on the requirement.

In 2012, Dr. Roumen Vesselinov and Dr. John Grego conducted Duoling's effectiveness research [12]. Real users with different backgrounds were involved to collect the data and effectiveness was measured in terms of an exam score taken before and after the online course completion on Duolingo platform. For the overall research, the average improvement of 91.4 points was noted before and after course completion. Though the effectiveness of this platform was coupled with the user's motivation, there was no mention of different exercises. Also, the focus of transfer learning between different exercises was largely out of the scope of the study.

The result from our experiment proves that 70%-80% of the knowledge is being transferred between different activities. This cross-learning experience cannot be compared to the study results by Dr. Vesselinov and Dr. Grego [12] as the focus of their study was the entire platform rather than independent activities. The methodology used by their research was dependent on real users spending time on the platform. They were able to measure 95% confidence interval for the effectiveness from every 5.6 to 10.7 points gain per one hour of exercises. Our report produces the result in line with this former finding.

Quantification of these activities provides the effectiveness measure users normally seek while trying to learn a new language. Online language learning platforms make these activities fun for users to quickly adapt and be addicted to. Small exercises and virtual rewards hook users to these platforms for long hours. This study is trying to answer the question of whether the user is learning anything by going through these exercises or it is just another online game-like experience without any gain.

For this report, first, we will go over the background of the technologies used in this project. Mainly including deep learning introduction and neural networks. The next section will go over the design aspect of the project. It will include design architecture, dataset processing, and environment used for this project. As multiple models are being used for conducting the project, the difference between each of them will be explained in detail. The section after that will provide information about the practical aspect of the project covering model training and testing results. Lastly, the conclusion for this project and possible future work is presented.

2 BACKGROUND

To provide the project context, we will look at the background technologies used in this project. First, we will review deep learning basics and look at the Neural Networks and RNNs. We will also review the sequence-to-sequence model for translating between two languages.

2.1 NEURAL NETWORK

The human brain is one of the most complex systems known to mankind [8]. Each human brain has billions of neurons transmitting and processing information. Each neuron has many inputs, a cell body, and a single output. Inputs are called dendrites and output is called axon [2]. Figure 3 shows the biological neuron and its terminology.

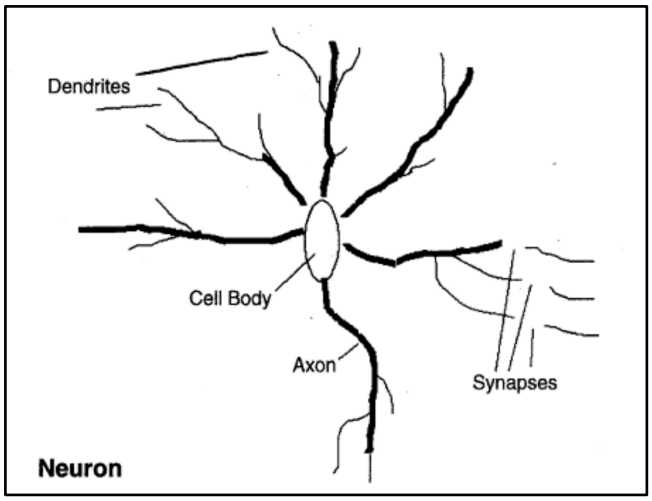

Figure 3. Biological Neuron. [8]

In artificial intelligence, perceptrons are the simple computational model of neurons. Perceptron has many inputs and one output. It is used for binary classification problems. As shown in figure 4, the inputs signals are modified with attached weights and fed into the processing unit. The unit sums all the inputs and use the transfer function to produce the

output. For example, a transfer function works with a threshold and if the input exceeds the threshold, the output is 1 else it is 0.

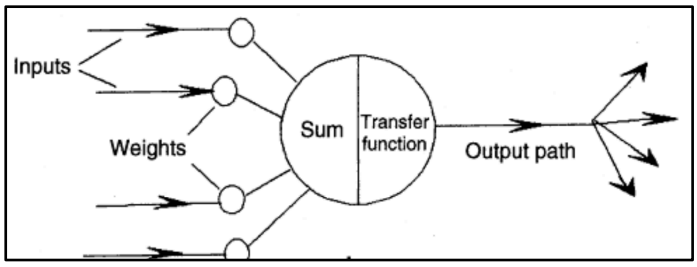

Figure 4. Schematic Diagram of a perceptron. [8]

Multiple perceptrons can be combined to solve the multiclass decision problem. This combination creates a linear unit with multiple perceptrons. Nowadays, the number of linear units are higher, and the computation happens in terms of layers. One layer is a group of units which either stores or computes information in parallel and passes it to the next layer. The word “deep” in deep learning technology refers to the piling of multiple layers [2].

Figure 5 depicts the structure of a typical neural network. Not all layers are connected at each stage to a single layer and this difference creates two types of the neural network, fully connected and partially connected. Data is provided at the input layer and response is produced at the output layer. Other layers in between are hidden layers. The number of inputs and outputs is decided by the problem set while the hidden layer neurons are decided by the program design.

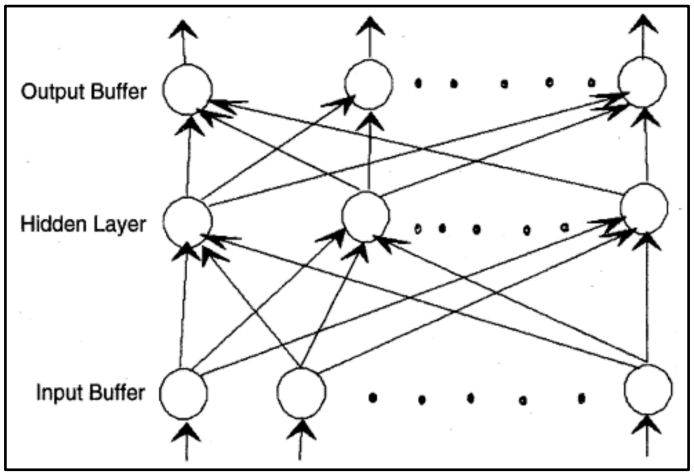

Figure. 5 Neural Network. [8]

The quality of the output can be measured by the loss calculation in the network. The goal of the neural network design is to minimize the loss. Loss function can be explained by the equation shown in Table 1 which implies that if we know the parameter impacting the loss, we should change those parameters to decrease the loss. Table 1 also illustrates the loss function graph which shows the loss function behavior over time. The threshold is one of the choices for the perceptrons to use for the transfer function. Systems other than perceptron use different transfer functions. Table 1 shows the equation for the Stochastic Gradient Descent function. Instead of working with vector values in the neural network output, a probability distribution can be used. The probability distribution is a set of numbers that are non-negative and sums up to 1. Softmax function can be used to convert vector output to the probability distribution.

Loss Function	$\Delta\phi_i = -\mathcal{L}\frac{\partial L}{\partial \phi_i}$

Softmax Function	$\sigma(\mathbf{x})_j = \frac{e^{x_j}}{\sum_i e^{x_i}}$
Derivatives of Stochastic Gradient Descent	$X(\Phi, x) = -\ln p(a)$ $p(a) = \sigma_a(\mathbf{l}) = \frac{e^{l_a}}{\sum_i e^{l_i}}$ $l_j = b_j + \mathbf{x} \cdot \mathbf{w_j}$

Table 1. Loss function, softmax function, stochastic gradient descent function [2]

In the multi-layered neural networks, the additional layers can have nonlinear computation between the layers. This nonlinear function is called the activation function. Rectified linear unit (relu), sigmoid, and lrelu are some of the activation functions. Table 2 shows relu and sigmoid activation functions.

relu	$\rho(x) = \max(x, 0),$
Sigmoid	$S(x) = \frac{e^{-x}}{1 + e^{-x}}$

Table 2. Activation functions (relu, sigmoid) [2].

To put everything together, a multi-layered neural network model can be defined as $Pr(A(x)) = \sigma(\rho(xU + b_U) V + b_V)$, where σ is a softmax function, ρ is a relu function, U and V are weights from first and second layer, and associated b values are their biases. The processing of inputs to loss calculation is considered a forward pass. This forward pass

information is passed in the backward pass to adjust weights of input to lower the loss. This technique is called backpropagation.

2.2 RECURRENT NEURAL NETWORK

There are multiple types of neural networks based on the problem requirement and dataset. Some of them are Feed-Forward Neural Network (FNN), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN). RNN utilizes the relationship between parameters at time t and parameters at time t-1. These characteristics allow the time dependencies of sequences [4].

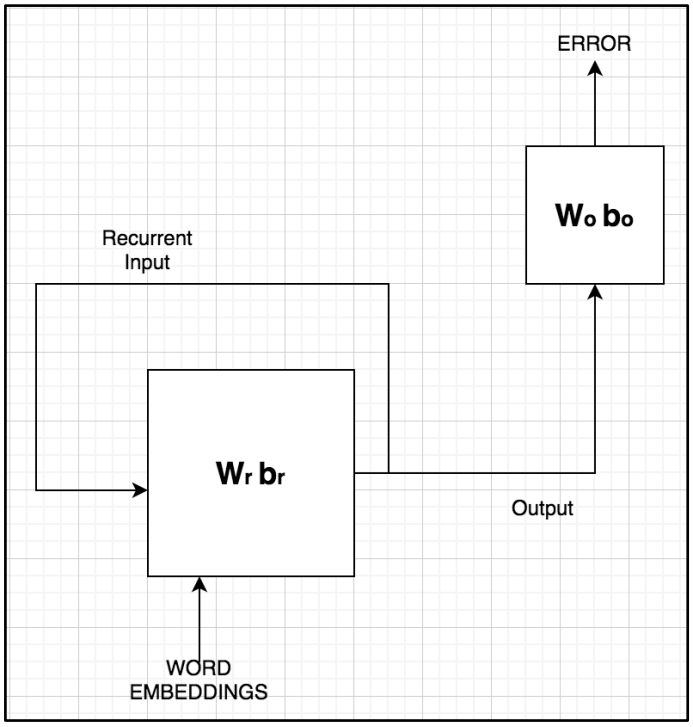

Figure 6 Recurrent Neural Network Illustration.

In the RNN, the output is used to improve the input. As shown in figure 6, the box with **W**r**b**r consists of linear units with weight **W**r and its related bias **b**r with an activation function.

The word embeddings are used as input to this box and the output is going to the next unit while a part of output goes back to the same neural network unit. This recurrent input gives the name of the recurrent neural network.

Here is the mathematical description of RNN cell shown in figure 6.

$$s_0 = 0$$

$$s_{t+1} = \rho((e_{t+1} \bullet s_t)\ \mathbf{W}r + \mathbf{b}r)$$

$$o = s_{t+1}\ \mathbf{W}o + \mathbf{b}o$$

S_0 represents the starting state and initialized as a vector of 0s. The next state is achieved by combining current input with the previous state and passing it through linear unit **W**r, **b**r. The output of RNN is **o** and is achieved by passing the current state through another linear unit **W**o, **b**o.

According to Liu, Wu, and Wang [7], RNN does have some drawbacks where information sequence cannot be analyzed on multiple time scales and gradient disappearance might occur while back-propagating in time. GRU can prevent gradient vanishing when the neural network is trained through backpropagation in time. GRU layer can also learn both long-term and short-term dependencies from sequences. In this way, the users' historical feedback sequences can be analyzed on multiple time scales. GRU has two gates: update and reset. Update gate is denoted as u and reset gate is r.

Here is the mathematical description for GRU from Zhang and Xiao [14].

$$u_j = \sigma(W_{ux}\ x_j + W_{uh}\ h_{j-1} + b_u)$$

$$r_j = \sigma(W_{rx}\ x_j + W_{rh}\ h_{j-1} + b_r)$$

$$h_j = \tanh(W_{hx}\ x_j + W_{hh}\ (r_j \bullet h_{j-1} + b_h))$$

$$h_j = (1 - u_j) \bullet h_j + u_j \bullet h_{j-1}$$

Here W and b are the learnable parameters of the system, h is the hidden representation vector at a given timestamp. X is the input vector at a given timestamp and is initialized with word embeddings.

2.3 WORD EMBEDDINGS

A language model is a formalization of the idea that one language sentence sounds different from the other [2]. The probability of words following the other words within a sentence also provides context for language modeling. The application of deep learning in language modeling requires us to convert words into an entity that can be manipulated. Word embeddings are the mathematical representation of the word.

There are two types of word embeddings: one-hot vector and distributed representation. One hot vector uses the index in the dictionary to represent the word uniquely. One-hot vector representation only separates word and do not carry the semantic meaning of the word. On the other hand, Juntian, Tao and Lin [3] mentioned that distributed representation uses a vector of real numbers and carries the semantic meaning of the word.

2.4 SEQUENCE-TO-SEQUENCE MAPPING

Sequence-to-sequence is a deep learning technique to map a sequence of symbols to the other when it is difficult to map on the individual basis itself. A typical application for the sequence-to-sequence program is computer language translation, also known as machine translation. Translation of the sentence in one language from the other cannot be done word by word basis in the computer as each language has its own grammar and different sentence

structure. The translation has to be left for the machine to decide and that's when the sequence-to-sequence paradigm comes into the picture.

Figure 7 shows the basic sequence-to-sequence learning model. The system is divided into two parts, the lower part is the encoder and the upper is the decoder. The encoder is fed with the source language sentence and the decoder outputs the target language sentence.

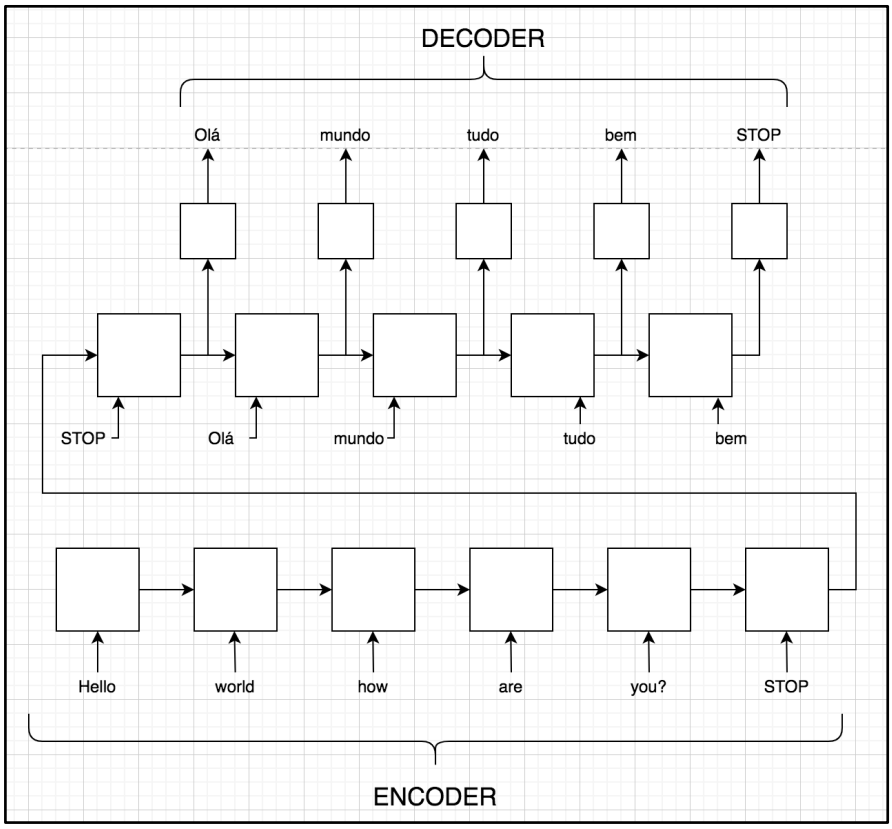

Figure 7. Sequence-to-sequence learning model.

There are two RNN cells, one for the encoder and the other for the decoder. Here the choice of RNN cell is done based on the problem requirement. Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) cells are popular choices. Each square box in encoder represents one single RNN cell through a memory line. The use of STOP at the end of the

sentence indicates the end of the encoding phase. The encoder cell iterates through the sentence and produces a summary of the sentence. The RNN cell state is then passed to the decoding phase. The sentence summary can be also described as sentence embedding, similar to word embeddings described earlier.

During the training phase, the decoder is provided with the target language sentence. At each stage in the decoder, the task is to predict the next word. As shown in figure 7, the decoder starts the process with word STOP and then proceeds to predict the next word. Again, each box shown in the decoder phase is a single RNN cell sharing a single memory line. Backpropagation in time is used for both RNN cells. The window size can be described as the amount of content the program can process at a given point in time. For backpropagation in time, the window size is a hyperparameter. This means that the user can decide the window size for the sequence-to-sequence process. Sentences in different languages can be of different sizes and may carry semantic information at different stages of sentences. For the same reason, the full sentence is used for both encoding and decoding phases. In-text corpus, the length of sentences may vary and require some kind of padding to keep the window size the same.

In figure 8, the encoder passes sentence embedding to the decoder. This embedding carries the meaning of the sentence. In source sentences, some part of the sentence could be more important than the other. This is not carried over to the decoder by the previously described implementation. Figure 6 shows the implementation of the sequence-to-sequence model with the attention mechanism. Every RNN cell output for the source language is summed up and passed to all RNN cells to provide the attention. The implementation in the figure shows

equal attention being given to all parts of the sentence. There is an option to provide attention to a specific position in the sentence and is called position only attention.

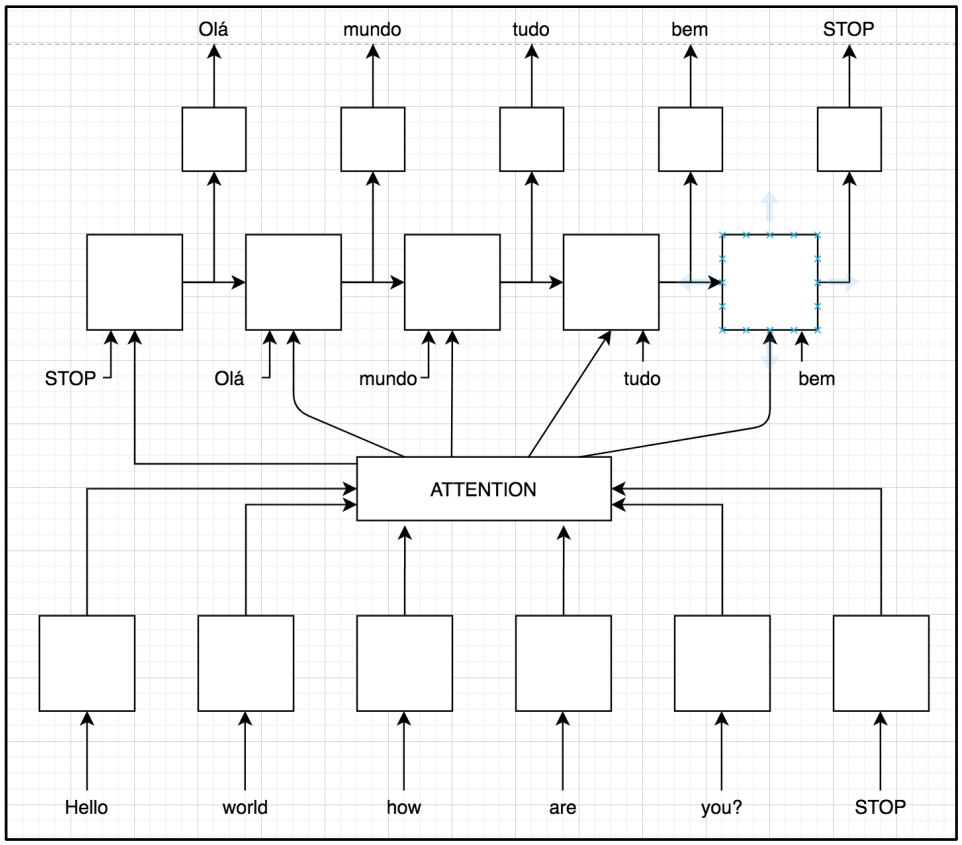

Figure 8. Sequence-to-sequence learning model with attention.

3 DESIGN AND IMPLEMENTATION

This section of the report covers the project design and implementation. It also has details about the dataset and its formatting. There were multiple models trained to generalize the language learning activities. Each model and its correlation to the learning activity is also explained in this section.

3.1 ENVIRONMENT

Python is used as a programming language for this project. It has a wide variety of open-source libraries available for machine learning processing. Specifically, this project uses the TensorFlow library for most of the machine learning work. TensorFlow version 1.15 has been used along with python 2.7. The data is preprocessed for the sequence-to-sequence model through the pickle library which helps in serialization and deserialization of python object. Dictionaries for source word to integer mapping and target word to integer mapping are stored in serialized format for the training phase. On the other hand, the same serialized data will be deserialized for the testing phase. Tensorboard is used to visualize the flow of the model and each component in the sequence-to-sequence model.

Google Collaboratory was also used to run experiments and collect data. It is a platform that provides online resources to run machine learning programs and store results. Google Colab also provides GPU to develop deep learning applications for free. It also supports python deep learning libraries such as TensorFlow, PyTorch, and NumPy for quick adaptation of the platform. To test the trained model, google translate was used to translate Portuguese sentences output by the model to English sentences.

Other python libraries used for this project are NumPy, pickle, and jellyfish. NumPy library is used to store encoder and decoder data in a file. This format keeps synchronization between English and Portuguese translation. Vectorized sentences in both languages are converted to NumPy arrays and stored in a single file as a two different dictionary. Pickle is the python library that can serialize and deserialize python objects. The data is converted to byte stream to store in a file and the same library can inverse it to convert byte streams to the object hierarchy. This function is used to store the English and Portuguese words to integer mapping which is used to create the vectorized sentences mentioned earlier. Jellyfish is the library used for phonetic and approximate matching of the two strings. To decide the efficiency of each model of this project, NumPy and Jellyfish libraries were used. Though Jellyfish supports different pattern matching algorithms like Levenshtein distance, Damerau Levenshtein distance, Jaro distance, Jaro-Winkler distance, and hamming distance. This project uses Levenshtein distance from this collection.

3.2 DATASET

There are two datasets used for this project. The first dataset has about 4000 English sentences followed by its Portuguese translations. This dataset was collected from the Duolingo research section [15]. There are multiple translations of a single English sentence each followed by weight. This attached weight corresponds to user response rates from the Duolingo platform. The first sentence followed by the English sentence is the most popular meaning in the Portuguese language. The rest of the sentences from the Portuguese language were used to collect random words. These random words were used in different quantities with different experiments.

The second dataset was from the Tatoeba project. It consists of approximately 123,644 English sentences and their translations in Portuguese sentences. Sentences in this dataset are sorted by length. The dataset is tab delimited. It has an English sentence followed by a tab that is followed by the Portuguese translation. Attribution information follows Portuguese sentences after a tab. This particular project does not use any information except for English sentences and their Portuguese translations.

Both datasets are combined to generate the data required for different experiments. Sentences are generated by combining English and Portuguese sentences in a different pattern for models.

```
**********************************************
***************DATASET  STATISTICS***************
**********************************************
TOTAL ENGLISH SENTENCES: 126443
UNIQUE ENGLISH WORDS IN DATASET: 24696
MAX ENGLISH SENTENCE LENGTH: 35
AVERAGE ENGLISH SENTENCE LENGTH: 6
MAX PORTUGUESE SENTENCE LENGTH: 33
UNIQUE PORTUGUESE WORDS IN DATASET: 36963
AVERAGE PORTUGUESE SENTENCE LENGTH: 6
**********************************************
***************SENTENCE STATISTICS***************
**********************************************
ENGLISH SENTENCES MORE THAN 30 WORDS: 5
ENGLISH SENTENCES MORE THAN 20 WORDS: 46
ENGLISH SENTENCES MORE THAN 10 WORDS: 5416
ENGLISH SENTENCES MORE THAN 0 WORDS: 120976
PORTUGUESE SENTENCES MORE THAN 30 WORDS: 4
PORTUGUESE SENTENCES MORE THAN 20 WORDS: 45
PORTUGUESE SENTENCES MORE THAN 10 WORDS: 6055
PORTUGUESE SENTENCES MORE THAN 0 WORDS: 120339
```

Figure 9. Dataset statistics

Figure 9 contains the dataset statistics used for this project. The final dataset contains a total of 24696 unique English words and a total of 36963 unique Portuguese words. The average sentence length in the dataset for both English and Portuguese sentences is 6 words. 95% of

the dataset sentences are less than 10 words in both languages. These numbers are useful to create models efficiently and also, they help with deciding sequence-to-sequence hyperparameters. This project also makes use of Portuguese random words which are collected from the datasets mentioned above.

3.3 ACTIVTIES AND DATASET PROCESSING

This project involves six different activities each imitating exercises provided by the Duolingo platform. The assumption is made that we are trying to learn the Portuguese language. Each activity has a different combination of English and Portuguese language sentences. Here are the different activities and their data processing workflows:

- **Write a sentence in a foreign language**: This activity provides the user with an English sentence. The user has to select Portuguese words in a correct order to translate the sentences. These Portuguese words are shuffled to ensure that the user understands the sentence structure in the Portuguese language. The same experiment can be used to generalize match-the-pair activity as well. This activity allows users to match a pair of words in English and Portuguese language.

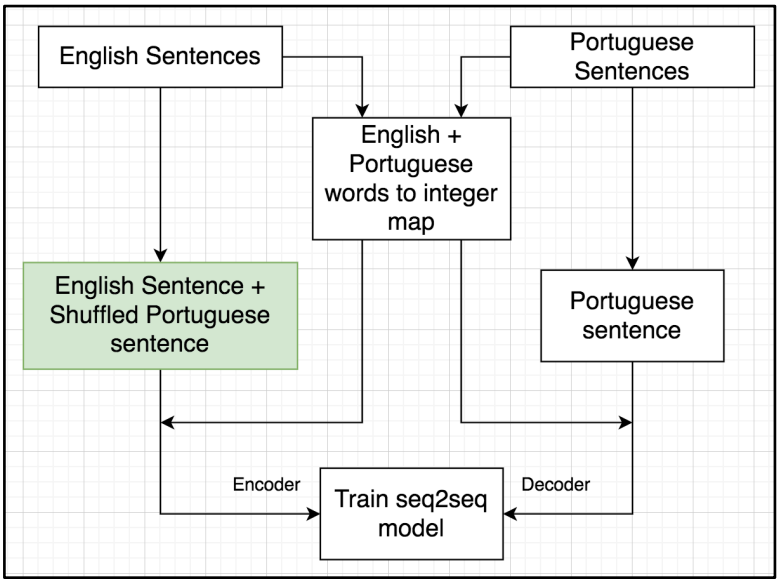

Figure 10. Data processing workflow for write a sentence in a foreign language activity.

As shown in figure 10, the English and Portuguese sentences in the dataset are split by the space to get all the words. All the unique words from this collection are given an integer value starting from 0. These mappings are stored in a pickle format so they can be used at a later time during training as well as the testing phase of the experiment. The model uses English sentences with shuffled Portuguese translation as an input to the sequence-to-sequence encoder. The exact Portuguese translation for the English sentence is fed to the decoder during the training phase. Here the string of words is converted to the list of integers from the word-to-integer mapping generated earlier.

- **Select the missing word to fill-in-the-blank**: This activity provides the user with a Portuguese sentence with one word missing. The user has to select the correct word to fill-in-the-blank.

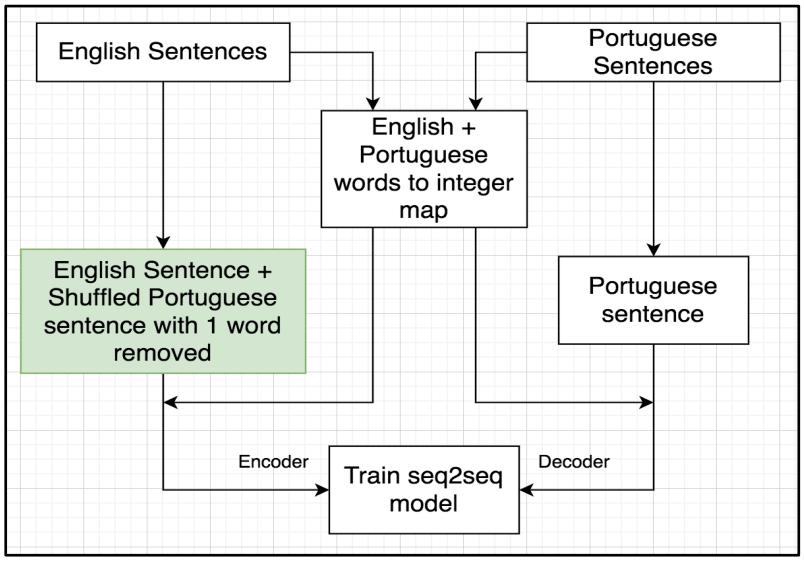

Figure 11. Data processing workflow for select the missing word in the blank activity.

Data processing for this activity is very similar to previous activity. As shown in Figure 11, the English and Portuguese sentences in the dataset are split to get all unique words from this dataset. The word to integer mapping is stored in a separate file like earlier activity to use it in testing as well as the training phase. The difference here is the encoder input. The English sentence and its shuffled Portuguese translation are combined except for removing one random word from the shuffle Portuguese translation.

- **Mark the correct meaning**: This activity provides the user with an English sentence and a few options in the Portuguese language to select the correct meaning. Except for the correct sentences, other Portuguese sentences are constructed with random words. In this experiment 10 random words have been selected from the Portuguese dataset. Figure 12 shows the workflow for this activity.

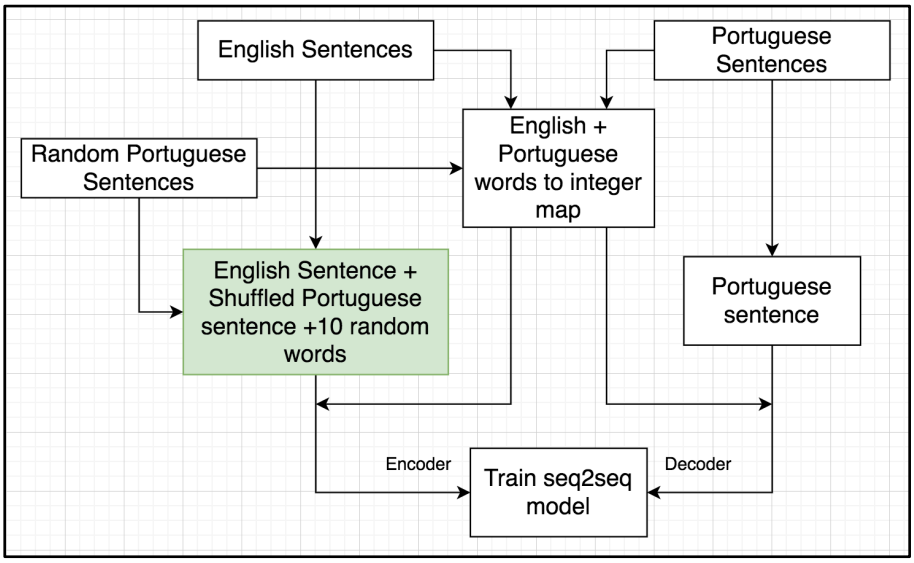

Figure 12. Data processing workflow for mark the correct meaning activity.

To mark the correct meaning activity, the difference is in the encoder input. A similar word to integer mapping is generated. This map will be stored and used to vectorize sentences for the testing and training phase. A separate dataset with random Portuguese words is used as a bag of words to select a specific number of words. The encoder will be fed with the English sentence along with shuffled Portuguese translation and 10 random Portuguese words from the word set mentioned previously.

- **Select the character**: For this activity, the user is given the English sentence and its Portuguese translation with a couple of extra words. The user is supposed to identify the extra words and get the correct Portuguese translation. Figure 16 depicts the workflow for this activity.

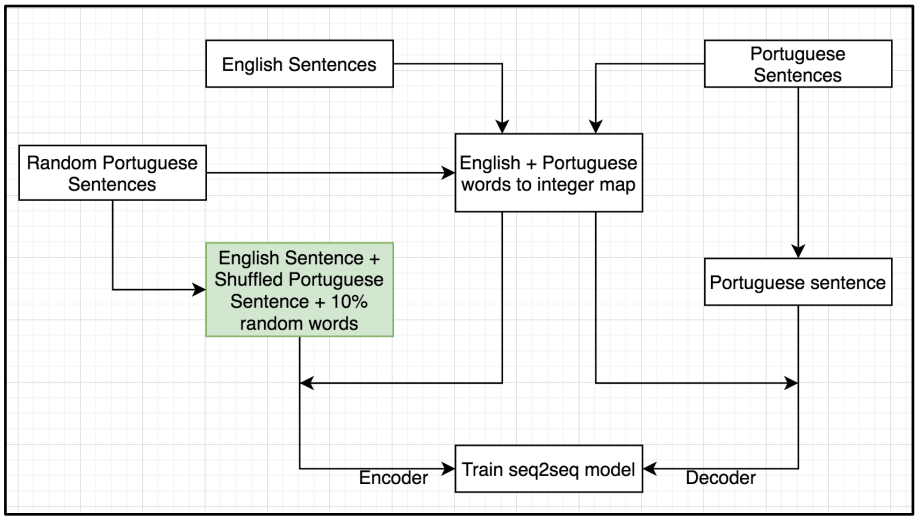

Figure 13. Data processing workflow for select the character activity.

As seen in figure 13, this activity is similar to all previous activities in terms to generate words to integer mapping. The main difference between the previous model is the number of random words. The number of random words used in this activity is limited to 10% of the number of words in Portuguese translation. As seen previously, 95% sentences in the dataset contain less than 10 words, so most of the sentences will be padded with only 1 random word.

As a variant for selecting 10% random words, another activity was designed to use 20% random words. This model focuses on the same aspects but with little more extra words as the activity does not have a fixed number of words. Figure 14 shows the workflow used for the same.

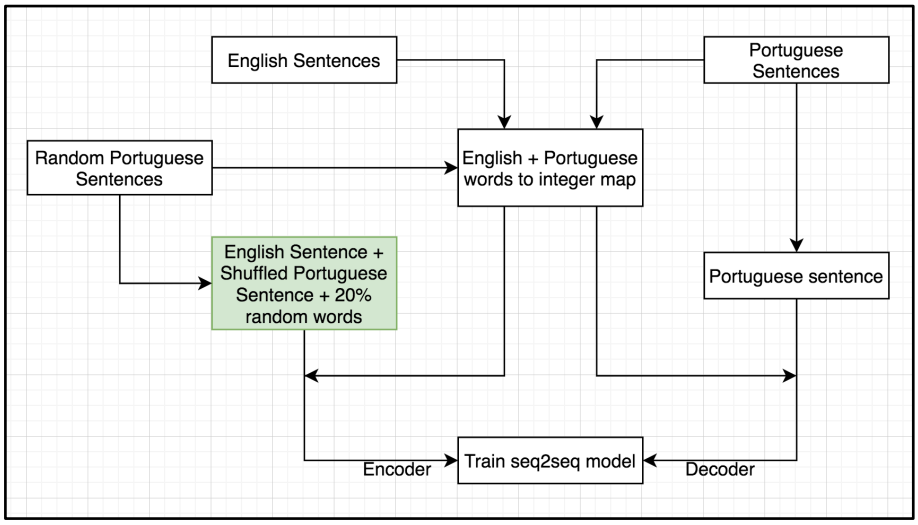

Figure 14. Data processing workflow for select the character training.

- **Sentence translation:** This activity was designed to prove the importance of other activities and their relevance. The ultimate goal for all previous activities is for the user to learn a foreign language. After going through all exercises on these online platforms, user expects himself to independently translate the sentence from one language to the other. This activity will imitate the user's intent to translate. Data processing is very similar to all previous activities. The difference here is the encoder input. No Portuguese translation is fed along with English sentences. Figure 15 shows the workflow for this activity.

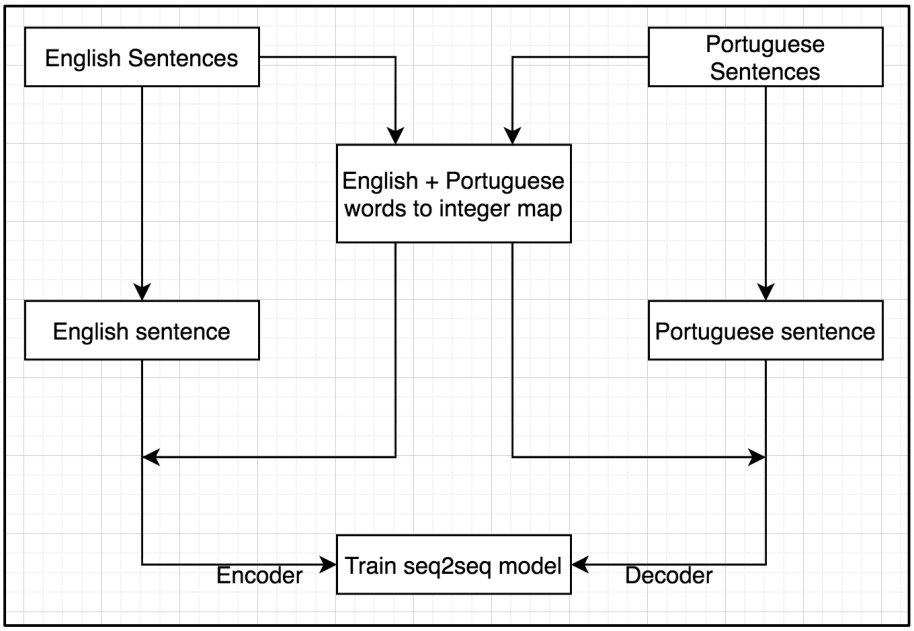

Figure 15. Model-6 workflow for sentence translation activity.

3.4 MODEL ARCHITECTURE

As mentioned in the previous section, the processed dataset for each activity is stored in two different files. One file contains the serialized data for an English word to integer and Portuguese words to integer conversion. The second file which is an input file contains vectorized sentences. The input file for the model is stored in the 2-dimensional array format with X-axis containing encoder input and Y-axis containing the decoder input. This information is fed into the word embedding layer which produces the word embeddings. Figure 16 shows the generic sequence-to-sequence model used for this project.

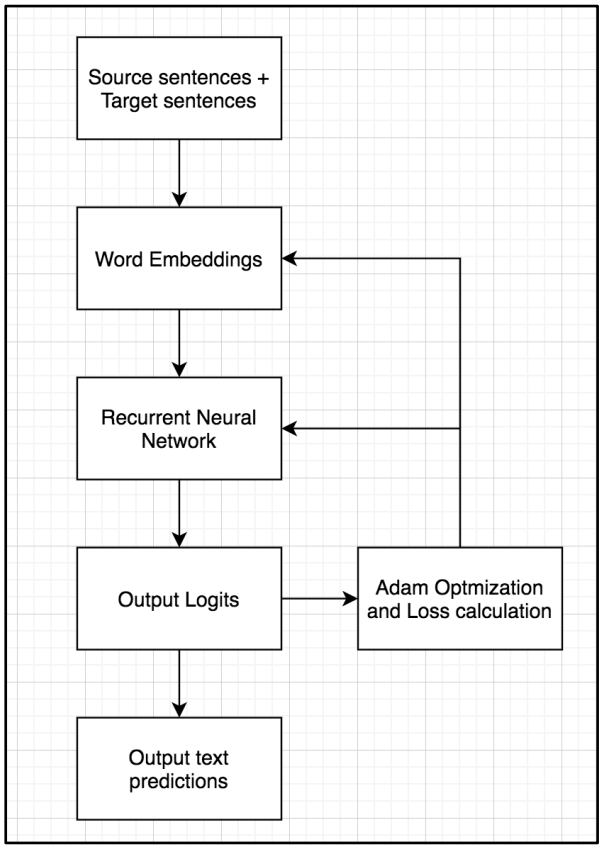

Figure 16. High level model architecture

The word embeddings are generated using the TensorFlow library which takes in the sequence of symbols and maps them to the word embeddings. The next layer is the multi-cell RNN which combines multiple single RNN cells, in this project GRU (Gated Recurrent Unit) cells, into a single entity. The next layer creates a neural network using the multiple RNN cells and the word embedding from previous layers. The output for the neural network layer is value and state. For the encoder part, the output values are ignored, and only states are passed on to the decoder. The decoder layer uses a helper function to read in the training values and pass it along with encoder states. These values are again fed into the recurrent neural network. The decoder layer outputs the values which are used for loss and accuracy calculation.

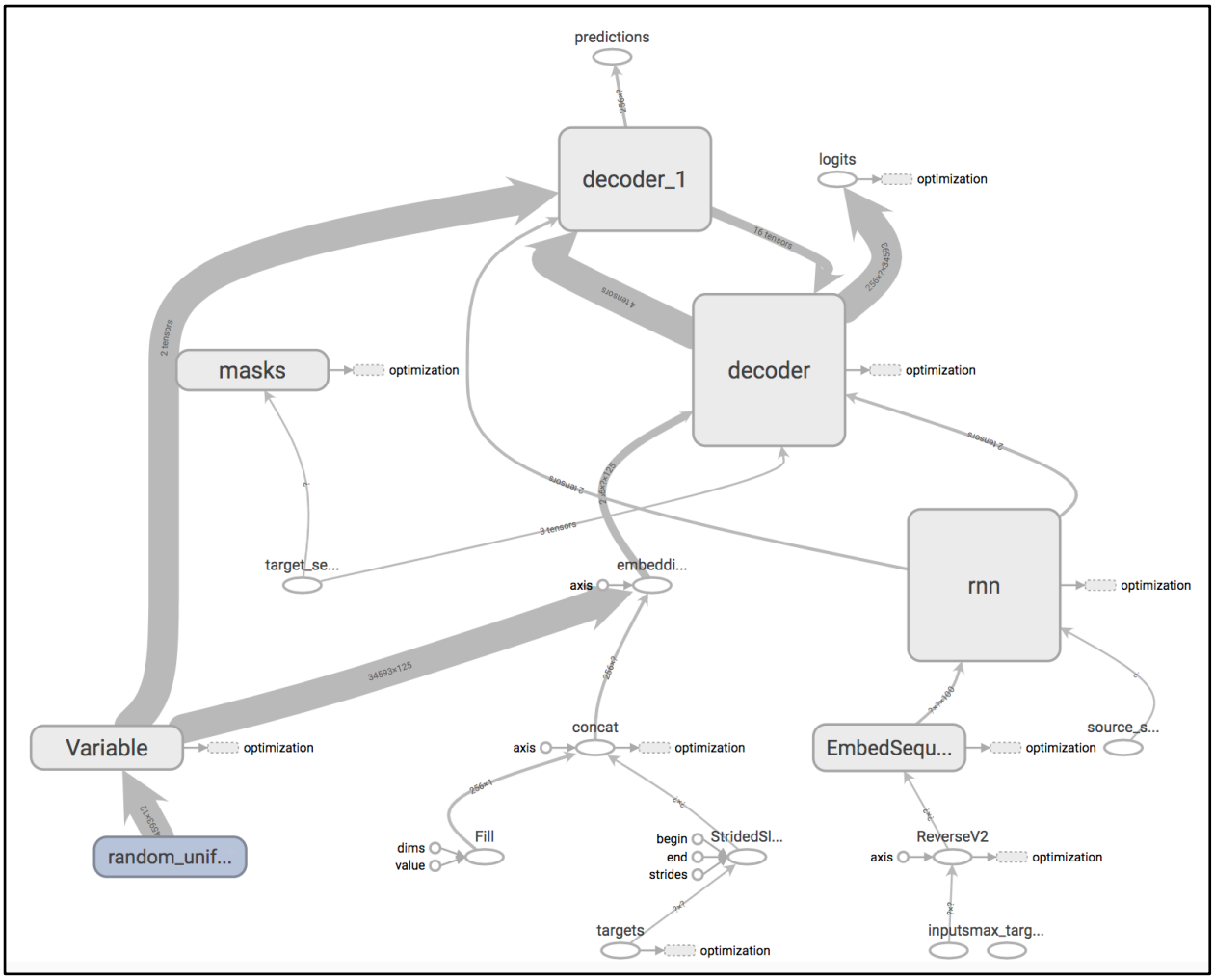

Figure 17. Tensorboard Diagram for the generic project model

The output values are also used in the optimizer. For this project, Adam algorithm optimizer is used. It is an extension of the stochastic gradient descent to improve the weights iteratively. Stochastic gradient descent maintains a single learning rate while Adam optimization has a per parameter learning rate. The loss function is a specialized version of cross-entropy loss used when logits are 3D tensors. It takes output logits, the correct answers, and weights to indicate some errors count more towards the loss than the others.

Tensorboard is a built-in tool in the TensorFlow library. It analyzes the logs and data captured through the training phase and uses these data points to plot a diagram. Figure 17 shows such a diagram for this experiment. As explained earlier, the input strings are fed to the embedding lookup box. That box sends the word embeddings to the RNN box as well as to the optimization. The output of the RNN box goes to the decoder which provides predictions as well as feed into optimization box for improvement of the result.

4 EXPERIMENTS AND RESULT

The dataset consists of a total of 126,443 unique English sentences and their Portuguese translations. As mentioned in the model explanation, there is a total of 6 defined models. The dataset was divided to generate training and testing data. The training dataset contains 85% of the sentences from the main dataset while the rest 15% is the testing dataset. All models share the same training dataset and testing dataset except for different sentence formatting for encoder input. The batch size for the experiment was set to 256 samples. RNN size, a hyperparameter that initializes the same number of units in the GRU cell is set to 128. A total of two GRU cells were initialized in the model. The total number of epochs was set to 40 and the learning rate was set to 0.001. The training for each model took about 24 hours on a personal laptop.

4.1 MODELS

Here we will map each activity defined previously to a sequence-to-sequence model. For each model, the difference is in the encoder input. The aim of this project was to have neural network models that can be used to generalize the language learning platform exercises. Though the scope of this project is not covering all available language learning exercises, it tries to cover the most generic form of exercises available on the Duolingo platform. The goal for the language learning platforms is for the users to learn a new language. Though each exercise expects a different kind of output, for the intention of generalizing the activity, decoder input was kept the same for all models. The difference is the input from the user for different exercises. Table 3 mainly describes the input value for the encoder for each sequence-to-sequence model. The mapping between encoder input and its matching exercise was already covered in section 3.

MODEL NUMBER	ENCODER INPUT
MODEL-1	English sentence + Shuffled Portuguese translation
MODEL- 2	English sentence + Shuffled Portuguese translation + 10 Fixed Portuguese random words
MODEL-3	English sentence + Shuffled Portuguese translation + 10% Portuguese random words
MODEL-4	English sentence + Shuffled Portuguese translation with 1 word missing
MODEL-5	English sentence + Shuffled Portuguese translation + 20% Portuguese random words
MODEL-6	English sentence

Table 3. Model Description for the experiment.

4.2 ACCURACY

Accuracy was calculated in two different ways to get more visibility for the project. The first way was to use python's NumPy library to get the original output and the expected output array in the same shape. Once they are in the same shape, the comparison is done element by element for each array. If the element is equal in both arrays, the library will return 1 on that position. An average is taken for each array and for all sentences in a batch. This is used as an accuracy.

The second approach was to use the Levenshtein distance. It measures the difference between the two strings. According to Apostolico and Galil, Levenshtein distance in a simple form between two strings would be the minimum number of character insertion and/or deletion

required to convert first string to the second [1]. A generalization of the Levenshtein allows the insertion, deletion, and substitution with their minimized respective costs.

There is a direct relation between Levenshtein distance, shortest edit sequence, and longest common sequence of the two strings [1]. If D is the Levenshtein distance between two strings of length l1 and l2, S is the length of the shortest edit sequence and L is the length of the longest common sequence, then S = D and L = (l1+l2-D)/2. Jellyfish python library is used to calculate this distance.

4.3 TRAINING

A total of 107,443 English sentences with their Portuguese translations were used to train these models. As mentioned earlier, epoch value was set to 40 with the learning rate set to 0.001. The loss for each model at the end of the training was plateaued. The total training time for each model was around 24 hours on a personal laptop. Model-6 has the lowest accuracy among all experiments and model-3 and model-2 have the highest. The loss was the highest for model-6 while model-1, model-2 and model-3 had the lowest loss number.

	Training Accuracy	Training Loss
MODEL – 1	98%	0.01
MODEL – 2	99%	0.01
MODEL – 3	99%	0.01
MODEL – 4	95%	0.03
MODEL – 5	98%	0.05
MODEL – 6	88%	0.16

Table 4. Training accuracy and loss from the experiment.

4.4 TESTING

Every trained model is tested with other model's test data. This is the exercise that will quantify the generalization of these six activities. Table 5 shows the testing result from each cross-validation experiment. The row represents the trained model and the column represents the test model.

		TEST MODEL					
		1	2	3	4	5	6
TRAINED MODEL	1	80%	53%	75%	72%	68%	55%
	2	70%	75%	71%	66%	73%	56%
	3	80%	57%	80%	70%	75%	55%
	4	72%	41%	63%	77%	56%	57%
	5	75%	61%	77%	68%	77%	55%
	6	29%	12%	23%	34%	20%	67%

Table 5. Testing results from the experiment.

For the testing part, all 19,000 sentences were kept the same across all models. The format of input for the encoder is different based on the model requirement. The batch size was kept at 256 so a total of 74 batches were run during the testing phase. The number in the table shows the average accuracy percentages during each experiment.

To verify the training as well as testing accuracy, each model was tested with a single sentence input in the expected form by model's encoder. Below is the testing output for each model with a single sentence. Figure 18 shows the testing output for model 1 where English

sentence with shuffle Portuguese translation was fed to the encoder and decoder produced the Portuguese translation. It can be seen that output is not entirely correct, but it does contain some of the words expected by the correct translation.

```
The size of English Map is : 35758
The size of Portuguese Map is : 35758
ENglish Sentence: "our teacher is very good muito bom professor nosso é"
INFO:tensorflow:Restoring parameters from checkpoints/model.ckpt
  Word Ids:      [2, 2848, 13736, 12297, 16580, 18304, 30503, 9153, 25657, 2]
  English Words: ['<UNK>', 'teacher', 'is', 'very', 'good', 'muito', 'bom', 'professor', 'nosso', '<UNK>']
  Word Ids:      [24987, 1947, 11878, 25657, 9153, 18304, 30503, 19424, 19543, 1]
  Portuguese Words: ['ela', 'pensou', 'em', 'nosso', 'professor', 'muito', 'bom', 'um', 'menino', '<EOS>']
ela pensou em nosso professor muito bom um menino <EOS>
```

Figure 18. Single sentence testing output for model 1.

Figure 19 shows the testing output for this model where English sentences with shuffled Portuguese translation along with 5 random words are fed to the encoder. Again, the output shows that most of the words from the correct translation are missing in the output. This irregularity can be attributed to the loss we see in the accuracy of model 2.

```
The size of English Map is : 35758
The size of Portuguese Map is : 35758
ENglish Sentence: "I'm alone. tarde, fica coisas. o vai a pro Estou com sozinho. autocarro pintora"
INFO:tensorflow:Restoring parameters from checkpoints/model.ckpt
  Word Ids:      [2, 2, 2, 26216, 2, 27318, 26744, 30246, 16415, 23182, 11480, 2, 7362, 2]
  English Words: ['<UNK>', '<UNK>', '<UNK>', 'fica', '<UNK>', 'o', 'vai', 'a', 'pro', 'estou', 'com', '<UNK>', 'autocarro', '<UNK>']
  Word Ids:      [9981, 10423, 3858, 32034, 6021, 13737, 23690, 16685, 6968, 29891, 1]
  Portuguese Words: ['as', 'senhoras', 'e', 'não', 'podem', 'ir', 'para', 'os', 'estados', 'unidos', '<EOS>']
as senhoras e não podem ir para os estados unidos <EOS>
```

Figure 19. Single sentence testing output for model 2.

Model 3 includes 10% random words along with shuffled Portuguese translation. Figure 20 shows the input English sentence and Portuguese translation had 4 words each. Because the sentence contained less than 10 words, a single random word was added. The accuracy seems to be about 75% because out of four, three correct words can be seen in the output.

```
The size of English Map is : 35758
The size of Portuguese Map is : 35758
ENglish Sentence: "they need your help eles ajuda de tua precisam"
INFO:tensorflow:Restoring parameters from checkpoints/model.ckpt
  Word Ids:      [2, 30906, 12481, 21288, 19313, 16101, 11424, 14245, 2]
  English Words: ['<UNK>', 'need', 'your', 'help', 'eles', 'ajuda', 'de', 'tua', '<UNK>']
  Word Ids:      [23591, 14245, 16101, 11426, 14245, 16101, 11424, 14245, 5596, 1]
  Portuguese Words: ['É', 'tua', 'ajuda', 'da', 'tua', 'ajuda', 'de', 'tua', 'irmã', '<EOS>']
É tua ajuda da tua ajuda de tua irmã <EOS>
```

Figure 20. Single sentence testing output for model 3.

Figure 21 shows the testing output for model 4 where English sentence with shuffled Portuguese translation sans a word was feed into the trained model. Here almost all words from the correct translation exist in the decoder output. It mimics the fill-in-the-blank activity.

```
The size of English Map is : 35758
The size of Portuguese Map is : 35758
ENglish Sentence: "i wont tell anybody you did that isso vou ninguém contar você que a não"
INFO:tensorflow:Restoring parameters from checkpoints/model.ckpt
  Word Ids:      [2, 11868, 13464, 17185, 30119, 31813, 29831, 10692, 1533, 965, 29904, 12054, 6305, 30246, 2]
  English Words: ['<UNK>', 'wont', 'tell', 'anybody', 'you', 'did', 'that', 'isso', 'vou', 'ninguém', 'contar', 'você', 'que', 'a', '<UNK>']
  Word Ids:      [30246, 12723, 12054, 32034, 26744, 29904, 30246, 965, 6305, 27318, 24779, 5168, 10692, 1]
  Portuguese Words: ['a', 'austrália', 'você', 'não', 'vai', 'contar', 'a', 'ninguém', 'que', 'o', 'tom', 'diria', 'isso', '<EOS>']
a austrália você não vai contar a ninguém que o tom diria isso <EOS>
```

Figure 21. Single sentence testing output for model 4.

5 CONCLUSION AND FUTURE WORK

This project explored the quantification of the generalization of language learning activities from online platforms. The neural network was used to achieve this quantification. The result shows that these activities can be generalized as most of the testing phases showed an average accuracy of 75%. The accuracy provides a way for us to know whether the knowledge acquired from one type of exercise can be utilized for the other. A similar level of accuracy proves that knowledge is transferable, and they are interconnected from the learning point of view. Also, the online learning platforms present the exercises in a specific order which was not maintained by this project. Different levels of expertise are also taken into account by these platforms in presenting these exercises. This is also omitted from the scope of this project. During the cross-validation phase, if we conduct testing in a specific order, we may see improvement in overall accuracy as the transfer of learning may increase between similar exercise concepts.

Lastly, the goal of this project was to understand the transfer of learning between exercise by generalizing the models. Model 6 was designed to highlight the importance of these exercises. Language learning is a difficult process. Translating sentences from one language to the other cannot be done correctly if the user is not aware of language structure and its grammar. All these exercises become ladder steps for the user to climb and reach the ultimate goal of learning a new language. Although according to Krashen, online language learning courses are based on conscious learning [6]. He claims that evidence suggests conscious learning does not produce language competence which aligns with the results from the last model in the project.

To conclude, though the result could have been improved by the use of attention mechanism which allows neural networks to provide more weight for a part of a sequence that matters more

than the other. Also, accuracy and loss for the model training can be improved by having more epochs on a powerful machine. The current experiments were done on a personal laptop and consumed a lot of time in each model training. The loss was not plateaued for any models which provide optimism for the improvement. Popular language learning platforms also have an audio exercise where users learn about dialects. This project does not cover any model which focuses on generalizing audio exercise but future work on this should be doable given this base experiment's promising results.

6 REFERENCES

[1] A. Apostolico and Z. Galil, *Pattern matching algorithms*. New York: Oxford Univ. Press, 1997.

[2] E. Charniak, *Introduction to deep learning*. Cambridge, Massachusetts ; London, England: The MIT Press, 2019.

[3] J. Chen, Y. Tao, and H. Lin, "Visual exploration and comparison of word embeddings," *Journal of Visual Languages & Computing*, vol. 48, pp. 178–186, 2018.

[4] V. Hernandez, N. Rezzoug, P. Gorce, and G. Venture, "Wheelchair propulsion: Force orientation and amplitude prediction with Recurrent Neural Network," *Journal of Biomechanics*, vol. 78, pp. 166–171, 2018.

[5] W. Y. (J. Ho, "Mobility and language learning: A case study on the use of an online platform to learn Chinese as a foreign language," *London Review of Education*, vol. 16, no. 2, pp. 239–249, 2018.

[6] S. Krashen, "Does Duolingo 'Trump' University-Level Language Learning? ," *The International Journal of Foreign Language Teaching*, pp. 13–15, 2014.

[7] J. Liu, C. Wu, and J. Wang, "Gated recurrent units based neural network for time heterogeneous feedback recommendation," *Information Sciences*, vol. 423, pp. 50–65, 2018.

[8] D. Mukesh, "Applications of Neural Computing for Process Chemists: I. Introduction to Neural Network," *Journal of Chemical Education*, vol. 73, no. 5, p. 431, 1996.

[9] S. Saini and V. Sahula, "A Survey of Machine Translation Techniques and Systems for Indian Languages," *2015 IEEE International Conference on Computational Intelligence & Communication Technology*, 2015.

[10] B. Settles and B. Meeder, "A Trainable Spaced Repetition Model for Language Learning," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016.

[11] "Tatoeba: Collection of sentences and translations.".

[12] R. Vesselinov and J. Grego, "Duolingo Effectiveness Study," 2012.

[13] R. Vesselinov and J. Grego, "Efficacy of New Language App," 2015.

[14] Y. Zhang and W. Xiao, "Keyphrase Generation Based on Deep Seq2seq Model," *IEEE Access*, vol. 6, pp. 46047–46057, 2018.

[15] Duolingo, “Data for the 2020 Duolingo Shared Task on Simultaneous Translation And Paraphrase for Language Education (STAPLE).” Harvard Dataverse, 2020, doi: 10.7910/DVN/38OJR6.