Abstract

Simultaneous Machine Translation (SiMT) is a branch of Natural Language Processing which tries to solve the task of translating streams of text from one source language to another target language in real-time. This is indeed different from the traditional machine translation system where the translator has the luxury of having complete information about a source text. Simultaneous Machine Translator, on the other hand has to make a decision at every time step, choosing either to get more context from the source or generate output prediction for the target. This incurs an interesting tradeoff between translation quality and latency, since waiting for more context from the source will potentially improve translation accuracy while incurring more waiting time. Recent works show that by incorporating visual information from images related to a source text, one is able to improve the translation quality of a plain SiMT system. This however, introduces extra latency from extracting image features, and combining both visual and linguistic context. Keeping these considerations in mind, the challenge for us is to build a SiMT system with visual context which has a fine balance between high quality and low latency. Finally, the aim of our project is to build a user interface that relies on a simultaneous machine translation model to provide a real-time translation service.
Acknowledgements

First, we would like to express our gratitude to the supervisor of this project, Dr. Lingpeng Kong, for his continuous supports, feedbacks, and encouragement throughout the development of this project. In addition, we are so grateful to have had Zhiyong Wu, our graduate supervisor, as a mentor who has provided help and insights during the first 4 months of this project. Last but not least, a sincere appreciation for Ms. Grace Chang and Ms. Mabel Choi for their guidance in consolidating and organizing our work in a much better way.
# Table of Contents

Abstract 2
Acknowledgements 3
Table of Contents 4
List of Figures 5
List of Tables 6
Introduction 7
  Consecutive & Simultaneous Machine Translation Systems 7
  Previous Work 8
  Goals and Objectives 11
  Project contribution & Significance 11
  Report Organization 12
Methodology 13
  Chapter Overview 13
  Dataset 13
  Baseline Neural Machine Translation 14
  Incorporating Visual Modality 15
  Enforcing Simultaneity 15
  Quality and Latency metrics 16
  Application 17
  Summary 18
Current Progress & Results 19
  Chapter Overview 19
  Project Status 19
  Challenges 21
  Future Plans 22
  Tentative Project Schedule 24
  Summary 25
Conclusions 25
References 26
## List of Figures

<table>
<thead>
<tr>
<th>Name</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fig 1.1 Consecutive Machine Translation</td>
<td>7</td>
</tr>
<tr>
<td>Fig 1.2 SiMT with Visual Context</td>
<td>7</td>
</tr>
<tr>
<td>Fig 1.3 Depiction of an ideal MMT interpreter</td>
<td>9</td>
</tr>
<tr>
<td>Fig 1.4 Depiction of policy based SiMT</td>
<td>10</td>
</tr>
<tr>
<td>Fig 2.1 Sample from Multi30K dataset</td>
<td>13</td>
</tr>
<tr>
<td>Fig 2.2 High level diagram of baseline CNMT model</td>
<td>14</td>
</tr>
<tr>
<td>Fig 2.3 High level diagram of baseline MMT model</td>
<td>15</td>
</tr>
<tr>
<td>Fig 2.4 SMMT model architecture</td>
<td>15</td>
</tr>
<tr>
<td>Fig 2.5 Formula for calculating BLEU score</td>
<td>16</td>
</tr>
<tr>
<td>Fig 2.6 Formula for calculating Average Lagging</td>
<td>17</td>
</tr>
<tr>
<td>Fig 2.7 Application Architecture</td>
<td>18</td>
</tr>
<tr>
<td>Fig 3.1 ResNet50 Model Architecture</td>
<td>19</td>
</tr>
<tr>
<td>Fig 3.2 Sample from Multi30K &amp; its feature matrices extracted from ResNet 50</td>
<td>20</td>
</tr>
<tr>
<td>Fig 3.3 Frontend interface screen capture</td>
<td>21</td>
</tr>
<tr>
<td>Fig 3.4 Transformer Architecture</td>
<td>23</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Name</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 3.1 N-gram precision of English to German translation task of the same model using different image encoder architecture</td>
<td>22</td>
</tr>
<tr>
<td>Table 3.2 Project Schedule</td>
<td>24</td>
</tr>
</tbody>
</table>
1. Introduction

1.1 Consecutive and Simultaneous Machine Translation Systems

Living in a multilingual society, the need for translation has a growing demand. Consecutive Machine Translation (CMT) systems have existed for years, and to some extent can provide a seamless experience with accurate translation. One major drawback, however, is the latency of translation experienced in such systems as the interpreter waits to collect the input text before it starts translating. Figure 1.1 depicts the working of a CMT system. The interpreter waits to collect the input text, here, “A red balloon”, which is then processed by the interpreter to output its French translation “Un ballon rouge”

Fig 1.1 Consecutive machine translation

The two root causes of latency in a CMT system, as shown in Figure 1.1, are waiting for the input text stream and the time utilized in the actual translation itself. In an effort to mitigate this latency, a “real-time” or “simultaneous” machine translation system is suggested, such that ideally we would be able to provide real-time translations to the input stream of text. Figure 1.2 depicts a theoretical picture of a SiMT model with the same input and output texts as used in the CMT model in Figure 1.1 above.
However, a simultaneous machine translation system presents challenges of its own, one major challenge being the need to build context for the interpreter [1]. The interpreter must progressively read a stream of input text and try to provide a real-time translation. Since the input text is being continuously fed to the interpreter, in order to provide low-latency during translation, there is a need to “anticipate” the next word [1]. For the human brain, this context could be built by a number of factors such as our linguistic ability, facial expressions, tone of voice, visual aid etc. Building the same context in a machine, however, is not as simple, and incurs latency cost of its own, as shown in Figure 1.2.

In this project since we are dealing with textual input, we focus on using visual aid to build lingual context for a machine interpreter. We refer to previously implemented SiMT models that use visual aid to build context and explore two possible pathways for our project. The first possibility is to build an application that could be used in real life to use SiMT with visual context for real-time translation in meetings, online conferences etc. Alternatively, optimizing the current state-of-the-art SiMT model by improving the latency measure and the quality of translation.

1.2 Previous Work

1.2.1 Multimodal Machine Translation

Multimodal machine translation or simply “MMT” attempts to improve the performance of an interpreter by drawing upon multiple modalities under the assumption that these additional modalities provide useful information [2]. As mentioned before, humans rely on a combination of visual, auditory and other stimuli being processed simultaneously to build complex relationships between objects that aid our quality of perception [2]. The theoretical idea then for a machine interpreter to achieve the same level of perception as a human brain would be to make use of not only a single isolated modality but to use different modalities together in harmony.
Referring to SiMT models that make use of visual cues along with textual input, the various available models differ primarily in feature extraction and also deciding when these features should be used during the process of translating.

Since the underlying goal for a MT model is to minimize the processing latency, minimizing the time required to extract features from the contextual image becomes of prime importance. Hence, to build such a model, one needs to carefully consider the trade off between time spent during processing the image vs producing the output text. Broadly speaking, the approach towards MMT using visual cues can be divided into two main categories. First being, interpreters that attempt to model an intra-sequential mapping between the input and output representations [6]. These interpreters lie under the family of “attention based decoders" which draw parallels between extracting certain image features and the alignment between the input and output text [6]. The second approach, on the other hand, makes use of an integrated attention-based neural network that works for both MT and image processing [7].

1.2.2 Simultaneous Neural Machine Translation

Referring back to the paper published by Cho and Espova in 2016, we take a closer look at a decoding policy that attempts to provide a prefix-to-prefix translation. This greedy approach known as “wait-k” can be seen as a policy-driven translation algorithm, where at each “state” or an “input word” the interpreter gets, the interpreter has 3 choices: whether to keep reading the next word, write the translation of this current word to the output or to predict the next word based on the contextual cues it has built.
Specifically for wait-k, “k” refers to the number of steps the interpreter reads for and then chooses whether to write or predict. So for such a policy, the interpreter is always k-steps behind the input stream of prefixes [5]. A benefit of this approach is that we can train our prefix-to-prefix SiMT model from scratch without relying on full-sentence models [5]. Amongst various policy based strategies, we can see that the basic intricacies involved in the decision making process are somewhat similar. We need to design a way in which we can decide at each state which action would yield the least latency without compromising the quality of the translation. To help make that decision, wait-k presents a deterministic solution where after every k-steps of reading the input text, we write or predict. Other than this however, we can explore other strategies that use heuristics or alignment-based approaches that could help in this decision making process [1].

1.2.3 Simultaneous Machine Translation with Visual Context

Simultaneous machine translation with an additional modality of visual cue to build context has been found to improve quality of translation under certain predetermined conditions [1]. Research has shown that SiMT models that use visual context perform considerably better, specifically, in cases of translating from English to gender-marked languages, for example, French, along with retaining structural integrity when it comes to different word-orders such as adjective-noun placements [1]. If we look at concrete metrics to judge the performance of a SiMT model that uses a “wait-k” policy, researchers have found that multi-models that use visual cues perform upto 3 BLEU score better than unimodal baselines. On top of this, it was found that using feature-based
visual aid can help resolve certain linguistic particularities such as gender marking and word order [1].

1.3 Goals & Objectives

There are 2 main objectives that this project aims at achieving,

1. To build an application that performs real-time translation between languages based on the current state-of-the-art SMMT model.
2. To do further exploration on MMT and SMMT with the hope of improving the current state-of-the-art in literature.

As an implication, not only will this project implement the existing SiMT models that use visual aid, but also use this existing technology to build a usable application that solves a real-life problem. More specifically, this project is moving towards building an SMMT plugin for existing web conference platforms such as Zoom, Microsoft Teams, Skype, and many more. Additionally, the project team is planning to also experiment with new ideas under SMMT setting to investigate if a better performance can be achieved. For example, by tweaking the model architecture or collecting a new dataset to train a baseline model on.

At the current state of this project, a baseline SMMT model mimicking the architecture of Caglayan et al. (2020) [3] has been built. A backend server coupled with a minimalistic frontend application has also been developed. The backend server was written in Python, while the client application was developed using JavaScript. On the research aspect, our team has experimented with various new image encoder architectures, and compared the performances. These experimental results will be discussed further in the latter section.

1.4 Project Contribution & Significance

As of the present moment, the area of simultaneous machine translation is of rising importance and is being increasingly sought after as a research topic. On top of real-time translation, the use of visual aid to build context not only adds complexity to the current task of translating with least possible latency, but also presents us with an interesting opportunity to delve into an area of SiMT models that have not yet been fully explored. Our project aims to contribute to this area of new and upcoming SiMT models that use visual aid as a feature to build context in hopes to mimic, even fractionally, the way lingual context is built inside a human brain. Ideally, we would like to help the current
state-of-the-art models to move a step closer towards providing a seamless translating experience. Our contribution would be in terms of suggesting ways of improving the current SiMT models that use visual aid, and also, if possible, trying to experiment with some of those suggestions to actualize their feasibility.

1.5 Report Organization

The report has been divided into five main chapters. The first chapter offers an introduction to machine translation, specifically simultaneous machine translation, along with some of the previous work that has been done in the field of building SiMT models that use visual aid to build context. It also serves as a platform for us to introduce the aim of our project to the reader and lay emphasis on the significance of our work.

Chapter two provides information on the methodology we plan to use during the course of this project. It gives a brief overview of the training dataset, the baseline model architecture we refer to and the various evaluation metrics used to evaluate the performance of SMMT models. On the engineering side, we will also briefly talk about the various technologies that will be used to build our application.

Chapter three is meant to inform the reader about our project schedule and provides a comprehensive list of deliverables we aim to finish.

Chapter four focuses on the current state of our project and discusses the results that we have achieved so far. This includes our initial SMMT model, and experiments on new image encoder architecture alongside with its performance. A minimalistic full-stack application that we have developed will also be discussed.

Finally, chapter five provides a succinct conclusion to this intermediate report that is meant to bind the arguments presented in the previous chapters to once again reiterate the significance of this project.
2. Methodology

2.1 Overview

In this chapter, we will discuss the dataset and evaluation metrics that were used for our experiments. In addition, we will also talk about the baseline SMMT model architecture that we experimented with. Finally, we will concisely walk through the problem that we had with the image encoder and the corresponding solution that we tried.

2.2 Dataset

For this project, we built our work based on the Multi30K dataset which consists of multilingual English-German image captions [9]. The Multi30K dataset is an extension of the Flickr30K dataset which contains 31,014 images from the world wide web [10]. Each image in the Flickr30K dataset is paired with a set of 5 independent English captions, generated using Amazon Mechanical Turk, a cloud-based crowdsourcing data labeling platform. The German translation of each captions were generated by professional human translators [9].

![Image](image.jpg)

1. Brick layers constructing a wall.
2. Maurer bauen eine Wand.

**Fig 2.1 A sample from the Multi30K dataset which consists of 1) source text in English 2) target text in German and 3) visual information associated with the texts**

Figure 2.1 shows a sample data point from the Multi30k dataset which is an image of construction workers constructing a brick wall accompanied by its English and German captions. With the availability of both visual and textual information, the project team believes that the Multi30K dataset fits the objective of building a state-of-the-art SMMT model. This argument is supported by the fact that some existing works have been done using this dataset by fellow researchers [1].
2.3 Baseline Neural Machine Translation

Neural Machine Translation is a task of Natural Language Processing that translates text from one language to another. This is different from SiMT (with visual context) in a few but significant ways. First, there is no need for anticipation since all information regarding the source text will be available prior to translation. This will eliminate the problem of latency as well. Second, there is an absence of multimodality, more specifically from the visual context side since visual information is made to be unavailable.

![High-level diagram of baseline CNMT model following an Encoder-Decoder architecture.](image)

However, one can think of SiMT (with visual context) as a modification or extension of NMT. With an additional enforcement of reading input source text progressively and information in the form of visual features. Hence, we will first discuss briefly on what would be the possible baseline NMT architecture for this project. From a very high level, a particular NMT architecture done in previous research consists of a 2-layer Gated Recurrent Unit (GRU) encoder and a 2-layer conditional GRU decoder with attention mechanism [1]. Think of an encoder as a black-box which converts input text into some abstract hidden representation, and a decoder as another black-box that generates output text given an abstract representation. To enforce simultaneity later on, the encoder shall be made unidirectional from left to right as input text will be progressively read.

We will start off by using this architecture as our baseline, and fine-tuning different hyperparameters as an initial experiment. If time permits, we are excited to dive deep into more details and tweak parts of our baseline language model as well.
2.4 Incorporating Visual Modality

Extending our explanation on baseline Neural Machine Translation. We will incorporate multimodality by adding visual information in a form of image feature tensor. This feature tensor will be extracted from the final convolutional layer of a ResNet-50 model, pretrained on the ImageNet dataset [11]. To combine this new visual information, an additional attention layer at each time-step of the decoder will be implemented for attending to the visual aid.

![Fig 2.3 High-level diagram of baseline MMT model built by adding a pre-trained Resnet-50 as image features extractor to the previous CNMT baseline](image)

2.5 Enforcing Simultaneity

Consecutive Neural Machine Translation, be it unimodal or multimodal, presents its own drawbacks, especially in the real-life application setting. For example, an automatic subtitle generator for web conferences. With CNMT, there is a high latency incurred by waiting for an input sentence to be available as a whole. This will cause a significant delay between the input speech and output subtitle which makes it infeasible for real-life implementations.

![Fig 2.4 Enforcing Simultaneity](image)
As depicted by Figure 2.4, instead of taking input text as a whole, SMMT accepts input on a word-by-word basis. This implies simultaneity as translation can happen at any time-step with an incomplete piece of information, hence enabling real-time translation. Our project team plans to follow a particular approach called wait-k decoding by Ma et al. (2019). This method reads the first k input words before generating the first output word [9]. It then reads and writes one word at each time step until the translation is complete.

2.6 Quality and Latency metrics

As mentioned in the previous chapter, there are 2 interrelated components of SiMT which can be optimized. First being, quality - how good or accurate the output text is. And another one being, latency - time delay between translation output and input text stream. To have a clear direction ahead, it is important to know how one can quantitatively measure these 2 very important metrics for SiMT tasks. Therefore, in this subsection, we would like to list out and briefly talk about different quality and latency metrics which will be used in our project.

2.6.1 BLEU Score

One of the widely used metrics for measuring machine translation quality is the infamous BLEU (Bilingual Evaluation Understudy) Score [12]. BLEU treats the quality of translation between one natural language and the other as good if it has high correspondence to professional human translators. BLEU score is then calculated on individual text segments - usually a sentence - by comparing it to a set of reference translations previously set as ground truth.

$$\text{BLEU} = \min \left( 1, \frac{\text{output-length}}{\text{reference-length}} \right) \left( \prod_{i=1}^{4} \text{precision}_i \right)^\frac{1}{4}$$

Fig 2.5 Formula for calculating BLEU score

BLEU Score ranges between 0 and 1. A BLEU Score of 1 represents a perfect match between generated machine translation and its set of references, while 0 means a totally deviating translation. In short, BLEU computes modified n-gram precision for various
window-length (in our case, we varied the window-length between the range $\varepsilon \{1,2,3,4\}$). It then takes the geometric mean of these modified n-gram precision. In the original paper, Papineni et al. (2022) [12] also mentioned about sentence brevity penalty which in short enforces a good candidate translation to match its references in terms of length, word order, and word choice.

### 2.6.2 Average Lagging

$$\text{AL}_g(x, y) = \frac{1}{\tau_g(|x|)} \sum_{t=1}^{\tau_g(|x|)} g(t) - (t - 1)$$

Fig 2.6 Formula for calculating Average Lagging

Average Lagging is a metric proposed by Ma et al. (2019) which estimates the number of tokens the generated output translation is lagging compared to the progressively read input text as a function of the number of source tokens has been read so far [5]. It is one of the latest translation latency metrics in the literature which we think will be useful throughout the project.

### 2.6.3 Consecutive Wait

Consecutive Wait is another latency metric that measures the number of source texts read between generating 2 consecutive translation tokens [13]. In short, we want to have a system with a low consecutive wait score which means fewer source context is needed for generating translations. In the SiMT with visual context setting, we expect visual modality can help adding extra information to anticipate reading too many of the source inputs.

### 2.7 Application

Our goal with the application is to demonstrate the potential uses of the simultaneous machine translation model in the real world. The use case we chose to target is the simultaneous translation of the speech in an online conference, meeting, or presentation. In those scenarios, we often have the presenter speaking while they share a video of themselves or an image from their presentation. To mimic that, we will build a minimal browser application using a JavaScript Framework (React) and several open-source libraries to be able to take in and process the presenter's video and audio. We will use the Speech Recognition interface of the Web-Speech API to convert the input audio to text
and take periodic snapshots of the speaker’s presentation/video to obtain the context image (if available). This text and image (optional) will be streamed into the server (Flask) through web sockets (Flask_SocketIO API). As the server receives the incoming data, it will use the Multimodal SiMT model to generate the translations of the text and stream it back to the frontend web application to be displayed.

![Figure 2.7 Application Architecture](image)

**2.8 Summary**

In this chapter we introduced the notion of performance metrics in regards to machine translation systems. We started with taking a look at the BLEU score which is used as a quality assessment metric in machine translation. Finally, we looked at Average Lagging which is a latency metric used to quantify the latency being incurred by a SiMT model.
3. Current Progress & Results

3.1 Overview

This chapter describes the progress we have made so far with the implementation of our SiMT model and the Simultaneous Translation server. It shares a big challenge we expect to face with the project in the future. The chapter finally looks at the Project Schedule and elaborates on the tasks we have in store for the future.

3.2 Project Status

3.2.1 PySiMT Library

So far, we have successfully replicated the results (BLEU score on the test sets) reported in the paper by Caglayan et. al. [1]. Along with their paper, they published the code they used in their research to the PySimt GitHub repository. The repository included implementations of several state-of-the-art Machine Translation models for both Consecutive Neural Machine Translation and Simultaneous Multimodal Machine Translation. We have thoroughly studied the code in the repository, noted issues and areas of improvement, and going forward, we hope to modify and optimize it for our purposes.

3.2.2 Image Encoder

To incorporate the visual modality, we make use of a pre-trained ResNet 50 model as the image encoder. The model architecture is shown below in Figure 3.1.

![ResNet50 Model Architecture](image)

Each image is represented by a feature tensor of dimensions $V \in \mathbb{R}^{2048 \times 8 \times 8}$. This tensor is extracted from the last convolutional layer of the model [1]. Currently we use this...
ResNet-50 model for object classification. There are 1000 distinct object categories available for classification of images. The ResNet 50 model consists of 48 convolution layers along with 1 layer for Max Pool and Average pool each.

![ResNet-50 model](image)

Fig 3.2 A sample image from Multi30K (left), and some of its feature matrices extracted from ResNet-50 image encoder (right)

As an example, Figure 3.2 shows a sample image taken from the Multi30K dataset alongside with its first 16 (out of 2048) feature matrices, each of size 8 x 8.

### 3.2.3 Backend Server

Once the simultaneous machine translation model worked end to end and performed sufficiently well, we isolated the code into a backend server. We modified the interface of the SiMT model to be able to take in single inputs sent over the network, instead of loading data in batches from a local file. We created a WebSocket server using Flask and the Flask_SocketIO api. It opens an endpoint to accept connections from frontend clients. These clients can now have a piece of text translated by streaming the input text and its corresponding context image (optional) through to the server. Once the server receives the stream of data, it calls the simultaneous translation function. The translation function passes the text and image into the model for translation and responds with the translated text.

### 3.2.4 Frontend User Interface

To test our backend server and demonstrate its capabilities, we created a dummy interface. We built a simple browser-based application - using React - that allows the user to upload an image and write some text. It uses the SocketIO library to connect to the
Flask SiMT Web Server, and stream the text and image to the server. Once the response is received from the server, the translated text is displayed to the user in the browser.

![SIMT Client](image)

**Figure 3.3: Frontend interface built using React JS**

3.3 Challenges

3.3.1 Dataset

To test our backend server and demonstrate its capabilities, we created a dummy interface. We built a simple browser-based application - using React - that allows the user to upload an image and write some text. This is then sent to the server in the body of a post request. Once the response to the request is received from the server, the translated text in the response body is displayed to the user in the browser.

3.3.2 Image Encoder

The model implementation from the PySiMT github repository does not provide us with the actual image encoder. Instead, it has a one-to-one mapping between each of the image files and the corresponding image feature tensors. This made it impossible to perform inference since for new images that fall outside the dataset, there is no way to encode it.

We studied the paper and tried to mimic the ResNet-50 image encoder implementation they had [1]. We were able to re-train a ResNet-50 on ImageNet and use that as a new
image encoder to generate new feature tensors. This is paramount, since with this, we will be able to use the encoder directly for new images during inference.

In addition, we experimented with new image encoder architecture called Inception_v3. Again, we generated new feature tensors for the training, validation, and test data with this new image encoder and retrained the sequence-to-sequence model using these new features.

<table>
<thead>
<tr>
<th></th>
<th>train 1</th>
<th>train 2</th>
<th>train 3</th>
<th>train 4</th>
<th>validation 1</th>
<th>validation 2</th>
<th>validation 3</th>
<th>validation 4</th>
<th>test 2016 flickr 1</th>
<th>test 2016 flickr 2</th>
<th>test 2016 flickr 3</th>
<th>test 2016 flickr 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Visual Context</td>
<td>73.5</td>
<td>62.55</td>
<td>54.42</td>
<td>47.7</td>
<td>64.8</td>
<td>51.6</td>
<td>42.2</td>
<td>34.79</td>
<td>64.7</td>
<td>50.9</td>
<td>41.3</td>
<td>33.7</td>
</tr>
<tr>
<td>ResNet-50 New</td>
<td>77.1</td>
<td>67.1</td>
<td>59.6</td>
<td>53.3</td>
<td>66.1</td>
<td>53.1</td>
<td>43.7</td>
<td>36.32</td>
<td>65.9</td>
<td>52.5</td>
<td>43.1</td>
<td>35.7</td>
</tr>
<tr>
<td>Inception_v3</td>
<td>75.6</td>
<td>65.11</td>
<td>57.18</td>
<td>50.6</td>
<td>66.1</td>
<td>52.9</td>
<td>43.5</td>
<td>36.14</td>
<td>65.3</td>
<td>51.9</td>
<td>42.4</td>
<td>35.1</td>
</tr>
</tbody>
</table>

Table 3.1 n-gram precision of English to German translation task of the same model using different image-encoder architecture (figures in bold are the maximum across column)

As shown in Table 3.1, the model when coupled with an Inception_v3 image encoder does indeed show improvement when compared to its counterpart without visual context. However, the performance is no better than when using the ResNet-50 image encoder. In addition, we ran an experiment encoding 1,000 random images using the two architectures and found that the average encoding time taken for ResNet-50 and Inception_v3 are 0.21s and 0.23s respectively. Not only does this imply that ResNet-50 is better in terms of quality, it also makes it more feasible for low latency applications.

3.4 Future Plans

3.4.1 Migrating from GRU to Transformer architecture
For further improvements, we believe that switching to the Transformer architecture would be beneficial. One drawback of using a GRU encoder is the vanishing gradient problem [14]. The problem stems from the fact that a GRU only takes into account the recent past through a series of hidden states. Hence, it is unable to learn which past input states might have a higher bearing on the current output. To mitigate this, we suggest
making use of the Transformer which relies on an attention mechanism [15]. The basic architecture of a transformer is shown below in Figure 3.2.

![Fig 3.4 Transformer Architecture](image)

In order to predict the next word in a sentence, the network looks at the input words that have been read so far. However, each of the previously read input words have a different value associated with them. Some words are more important than others. A direct consequence of this approach is that the word at time step (t-1) may no longer be the most valuable while predicting the output at time step t [15]. With the attention mechanism, we are able to pay less attention to the less relevant words and more to more relevant ones.

3.4.2 Multiple Languages

Presently, our models only support English to German and English to French models. We would like to enable the translation of a wider range of languages with the application. As such, we hope to collect more data that maps text between a wider range of languages.

3.4.3 Better Data

On the topic of data, we have previously mentioned the problems with the available image datasets. In the future, we plan to collect and/or create better quality datasets that will allow us to train a more general Simultaneous translation model.
3.4.4 User Interface

To have a practical application that can be used for real-time translation, it is necessary to gather the user’s video and audio stream. As such, we must integrate libraries that allow us to enable such features. These inputs must be further processed for efficiency, e.g., the audio stream must be converted to text using the Web-Speech API. The processed input will be streamed to the server via web sockets for translation. These updates will bring us closer to the application described in section 2.7.

3.5 Tentative Project Schedule

Here is the tentative schedule for our project for the next couple of months until the FYP submission deadline. We have already finished the inception phase of our project by building a basic React frontend to accept the input & a Flask server to respond to requests. We are currently working on implementing web sockets to enable real-time communication between the frontend and backend instead of using REST API calls. We are also experimenting with different image encoders to see if changing to a different architecture provides an improvement in the quality of translation.

<table>
<thead>
<tr>
<th>Date</th>
<th>Task Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 21’</td>
<td>Develop Website &amp; Project Ideation</td>
</tr>
<tr>
<td></td>
<td>Deliverable 1</td>
</tr>
<tr>
<td>October 21’</td>
<td>Reproduce Results from original paper</td>
</tr>
<tr>
<td></td>
<td>Read related background research</td>
</tr>
<tr>
<td>November 21’</td>
<td>Finalized direction of application path</td>
</tr>
<tr>
<td></td>
<td>Set goals for the rest of the project</td>
</tr>
<tr>
<td></td>
<td>Create rough sketch of application design</td>
</tr>
<tr>
<td>December 21’</td>
<td>Create frontend and backend interfaces</td>
</tr>
<tr>
<td></td>
<td>Find areas of improvement for current model</td>
</tr>
<tr>
<td></td>
<td>Experiment with different parameters</td>
</tr>
<tr>
<td>January 22’</td>
<td>Presentation for mid term review</td>
</tr>
<tr>
<td></td>
<td>Creating web socket interface</td>
</tr>
<tr>
<td></td>
<td>Experimenting with different image encoders</td>
</tr>
<tr>
<td></td>
<td>Deliverable 2</td>
</tr>
<tr>
<td>February 22’</td>
<td>Finish backend development</td>
</tr>
<tr>
<td></td>
<td>Model, API, server, cloud &amp; deployment</td>
</tr>
<tr>
<td>March 22’</td>
<td>Finish frontend integration</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>April 22’</td>
<td>Finish development &amp; soft launch</td>
</tr>
<tr>
<td></td>
<td>Final presentation</td>
</tr>
<tr>
<td></td>
<td><strong>Deliverable 3</strong></td>
</tr>
</tbody>
</table>

Table 3.2 Project Schedule

3.6 Summary

To summarize, we have reproduced the results reported by Caglayan et. al. [1] and used their SiMT model to build an initial version of our application. It supports real time communication between the server and the client, allowing simultaneous machine translation. While this is good progress, we have significant challenges to overcome and several tasks to complete.

4. Conclusions

In conclusion, the aim of this project is to build upon the existing SiMT models that take advantage of multi modalities. In our case, we explore the use of visual aid to complement lingual context. We plan to investigate some of the presently available successful strategies such as wait-k, alignment-based heuristics, etc. and try to understand the intricacies behind the working of these algorithms. Our aim is to build an application that utilizes a SiMT model with visual context to provide a real-time translation service.
5. References


