Advanced Techniques in Hindi Automatic Post-Editing: Neural Models and Data Augmentation

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computational Linguistics by Research

by

Pranav Nair
201525149
pranav.nair@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500032, INDIA
June 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “Advanced Techniques in Hindi Automatic Post-Editing: Neural Models and Data Augmentation” by Pranav Nair, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Advisor: Prof. Dipti Misra Sharma

Acknowledgments

My journey through Computational Linguistics was not one I walked alone. I'd first like to thank my school's Computer Science professor - Mr Arul Das - for introducing me to the field of Natural Language Processing. My introduction to NLP and NLU during school is what propelled me to join under this CLD dual degree program.

My path of research has not been easy. I've had to go through the entire process of it, from finding the right problem to finding a working solution that offers an advantage over every previous work - and have dabbled in trying to solve multiple different problem statements. I've had to scrap the work I did for over 3 years and pick up a new problem statement. I am glad that I was given the opportunity to face so many challenges, they've certainly shaped me into a stronger and more efficient student and human being. I would like to express my profound gratitude to my advisor, Dr Dipti Misra, for all the patience she has shown as well as the freedom she provided that allowed me to explore contrasting corners of NLP research. I thank her for giving me the freedom to also work on problems that have required an amalgamation of knowledge and practical information in the multimodal space. She supported me in working on a multimodal problem, despite the demanding expertise and learning required in Computer Vision.

Implementing that solution correctly, seeing it fail, and repeating this till you find something that works - and then writing a paper on it, seeing it rejected by conferences, fair or unfair, making improvements, and submitting again till it's accepted - it's quite stressful, and takes months if not years. I'm very thankful to the close friends I made in college who have kept the stress and anxiety from getting to me and the wonderful professors at LTRC who have helped shape my career. From not knowing the difference between syntax and semantics to working on Hindi for Microsoft Cortana as a client for my data science consultancy, they have helped shape my future in a very major manner. I would also like to extend my gratitude to Mr Arafat Ahsan for being a guide and a mentor and for taking time out multiple times every week to review my work, give suggestions, and help me progress.

I would once again like to thank Dr. Dipti and Mr. Arafat for their understanding of my health challenges and for their remarkable patience with me. I can never thank them enough for the same.

I'd also like to thank Pranav Goel, Rohan Tiwari, Siddhant Srivastava, Sri Aurobindo, Aniruddha Deshpande and all other batchmates, seniors and juniors who have helped me out and acted as advisors. I'd also like to thank my parents for giving me the liberty to pursue my career and academics at the same time.

My experiences and learnings in my research journey have been invaluable. They've shown me that there's a lot more to explore and learn, and have given me new perspectives on how to approach different subjects. I'm sure I'll apply these learnings no matter what I seek to do in life.

Abstract

Automatic post-editing (APE) is a crucial technique for enhancing the quality of machine translations. In this thesis, we present an APE approach specifically for English-Hindi translation. Our method employs a sequence-to-sequence neural machine translation model to generate initial translations, followed by a combination of neural and data-augmentation post-editing techniques to refine these translations. We evaluate our approach using a large-scale dataset of English-Hindi translations and demonstrate significant improvements in the quality of the initial translations, as measured by standard automatic evaluation metrics such as BLEU, CHRF, COMET, and TER. Our analysis further reveals that our approach effectively corrects specific errors commonly made by machine translation systems in the English-Hindi language pair, such as incorrect word order and grammatical agreement. We experiment with both neural models, data augmentation, as well as a mix of both to derive an ensemble model that works best for this problem statement. These results highlight the effectiveness of different APE approaches and their potential to substantially improve the quality of machine translation outputs for this language pair. Different evaluation metrics in the study show us various nuances of the functioning of the MT and APE models. These nuances are not only observed but are also analyzed and explained, providing deeper insights into the strengths and limitations of our approach(es).

Contents

List of Figures

Figure Page

List of Tables

Chapter 1

Introduction

Automatic post-editing (APE) refers to the process of automatically correcting and improving machine translation output to make it more fluent and accurate [3]. This tool is useful for overcoming the limitations of machine translation, which often struggles with complex grammar, idioms, and cultural references [4, 5]. As studied by Parton et al [6], human evaluation shows that the APEs significantly improve adequacy, regardless of approach, MT system or genre: 30-56% of the post-edited sentences have improved adequacy compared to the original MT [7]. In this study, we will focus on the challenges and approaches to developing APE systems for the Hindi language. While parallel data is fairly easy to come by, even for more unfamiliar languages, the unavailability of triplet data for automatic post-editing purposes has led us to stick with Hindi due to this limited availability of other Indian languages.

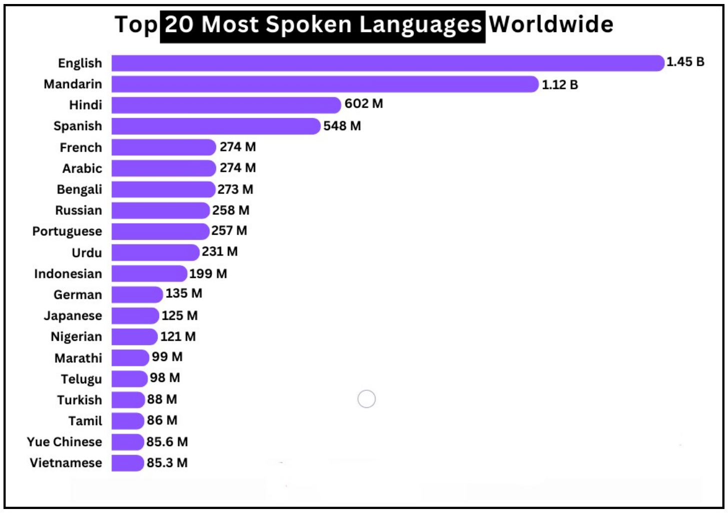

Figure 1.1: Languages spoken around the world [1]

Hindi is a widely spoken language in India and Nepal, with over 500 million speakers worldwide. It is written in the Devanagari script and belongs to the Indo-Aryan language family. Despite only being spoken by 53% of Indians, it still accounts to be the third most spoken language in the world. Hindi has complex grammar with a large number of inflections, as well as a rich literary tradition with a wide range of registers and styles [8]. The grammar placement of subjects, predicates, verbs, and nouns is significantly different than in English. Languages can be differentiated in terms of structural divergences and morphological manifestations. English is structurally classified as a Subject-Verb-Object (SVO) language with a poor morphology whereas Hindi is a morphologically rich, Subject-Object-Verb (SOV) language. Largely, these divergences are responsible for the difficulties in translation using a phrase-based/factored model, which we summarize in this section [8]. These features make it a challenging language for machine translation and APE systems.

One major challenge in developing APE systems for Hindi is the lack of parallel data, which is necessary for training machine translation models. While there are a number of resources available for machine translation of Hindi, such as the HindEnCorp dataset [9] and IIT-B Parallel Dataset [10] (which will be used later on in this study as the dataset for APE later on), these datasets are often limited in size and quality and most importantly, are not in the triplet format required to solve Post-Editing. This can make it difficult to train accurate machine translation models and improve their output through post-editing.

To address these challenges, researchers have explored a number of approaches to developing APE systems for Hindi. One approach is to use machine learning algorithms to automatically correct errors in machine-translation output. These algorithms can be trained on large amounts of parallel data to learn how to identify and correct errors in translation. Another approach is to use dictionaries and grammar rules to identify and correct errors in machine-translation output. This can be particularly useful for correcting errors in inflections and grammatical constructions.

In conclusion, the development of APE systems for Hindi faces several challenges due to the complexity of the language and the limited availability of parallel data. However, researchers have made significant progress in developing APE systems for Hindi through the use of machine-learning algorithms, augmented data, dictionaries, and grammar rules. Further research in this area is necessary to continue improving the accuracy and fluency of machine translation for Hindi.

[1] https://keywordtool.io/blog/most-spoken-languages-in-the-world/

1.1 What is Machine Translation?

Machine Translation in Natural Language Processing (NLP) refers to the task of using statistical, rule-based or neural methodologies to convert text from one language into another. In the case of this study, we will focus only on English-Hindi Machine Translation.

Neural Machine Translation (NMT) systems have made significant strides in producing translations that are contextually richer and sound more natural compared to older statistical methods [11]. The improvements in Google Translate's English-to-Hindi translations are a testament to this progress, demonstrating proficiency in handling day-to-day sentences. This advancement is largely due to the extensive parallel corpora that train these NMT models, allowing them to better understand intricate grammatical structures and idiomatic phrases [10]. Despite these advancements, several challenges remain. For instance, the complex structure of Hindi, a fusional language where single affixes can express multiple grammatical categories, continues to be a significant obstacle [12]. Misinterpretations of morpheme boundaries can lead to erroneous translations. Furthermore, the presence of homophones and words with multiple meanings (polysemy) can cause confusion for these models, leading to ambiguous or semantically incorrect translations. Another major challenge is the translation of domain-specific content, such as legal, medical, or technical texts [13]. These areas require specialized knowledge that most general MT systems lack, which can result in dangerously incorrect translations.

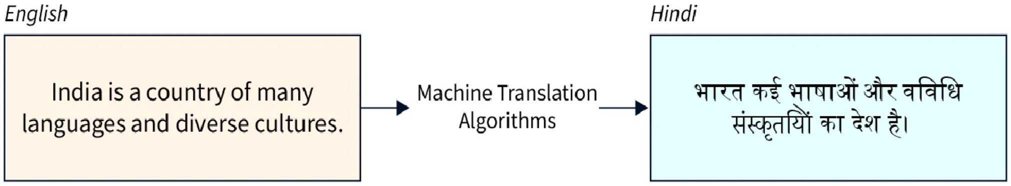
English
India is a country of many languages and diverse cultures.
Machine Translation Algorithms
Hindi
भारत कई भाषाओं और वविधि संस्कृतयिों का देश है।

Figure 1.2: Example of Machine Translation[2]

[2] https://analyticsindiamag.com/sequence-to-sequence-modeling-using-lstm-for-language-translation/
.

Current research efforts are addressing these challenges. The emerging field of multimodal NMT, which combines visual or auditory information with text, is showing potential in enhancing translation quality, especially for languages with limited resources like Hindi [14, 15]. Pre-training models on extensive monolingual Hindi datasets also appears to be an effective strategy for improving the models' grasp of linguistic nuances [16]. However, despite these technological advances, the role of human intervention, particularly through automatic post-editing, remains critical. Human oversight is necessary to spot and correct errors related to morphological complexity, ambiguities, and specialized terminology. The human touch is also essential for preserving the intended meaning and stylistic elements in creative content.

Therefore, automatic post-editing tools are vital in the realm of English-to-Hindi machine translation. By employing methods such as error detection, extraction of domain-specific terminology, and style transfer, these tools can significantly enhance the quality and accuracy of translations. This enables human translators to concentrate on the more subtle aspects of language, thereby fostering better cross-cultural communication and understanding.

1.2 What is Automatic Post Editing?

Automatic post-editing (APE) in natural language processing (NLP) refers to the process of using machine learning algorithms or rule-based systems to automatically correct errors or improve the quality of machine-generated text output [17]. It is an area of research aiming at exploring methods for learning from human post-edited data and applying the results to produce better Machine Translation (MT) output [4]. When generating text using NLP models, the resulting text may contain errors, such as grammar or spelling mistakes, or may not sound natural. Machine Translation has been a well-explored area by researchers for over 5 decades [18] and while this problem tackles the problem of representing the semantics of text in a given language to another, the problem of correction of translated linguistic content - be it editing the style and syntax of the linguistic content or to better improve the quality of translation given a specific domain - is a strong requirement.

The goal of APE is to improve the output of machine translation systems, based on the knowledge extracted from datasets that include post-edited content [4]. This process can be achieved through various methods, including statistical machine translation, neural machine translation, or rule-based systems. The goal is to produce high-quality text that is grammatically correct, semantically accurate, and sounds natural to a human reader [19].

Without APE as a problem statement, a lot of translated data would have to be manually edited. Additionally, while correcting large pieces of text can take multiple humans multiple hours to complete, a machine-run model can do the same at a cheaper and much more

time-efficient manner. Additionally, an APE model can be trained to be more accurate than a human reader in identifying errors, ensuring that text is accurate and free of syntactic mistakes [20]. Speaking of syntax, APE can help maintain consistency in grammar, spelling, punctuation, and other style conventions. Not all post-editing is done to correct syntax, multiple times - based on factors like domain and tense, APE can help correct the meaning of a piece of text too.
Overall, automatic post-editing can help produce high-quality content quickly, efficiently, and cost-effectively, which is essential for individuals and businesses who rely on content marketing to grow their audience and reach their goals.

1.3 Critical Challenges in Automatic Post-Editing

NLP is a complex and challenging task - the automated system must have the ability to analyze the text, identify the relevant parts of speech, and understand the meaning of words in context. Additionally, the black-box nature of end-to-end machine translation systems makes it difficult to understand how source language inputs are being mapped to the target language [21, 22].

Similarly, a big reason for the language being a complex entity is due to its multiple moving parts that need to be taken into consideration to learn or define said language. Understandably, sentences with more complex structures are more difficult to understand, translate, and edit for humans. They must understand the intended meaning of the writer and the underlying nuances in language, idioms, sarcasm, or irony used in the post. It is notoriously difficult to evaluate these systems and demonstrate that they are safe to be used in a clinical setting [23]. The style and tone of a post are essential for conveying the writer's intended message. An automated post editor must be able to identify and adjust the tone and style of the text to match the writer's intended tone.

Automatic post-editing requires knowledge of the rules of grammar, syntax, and sentence structure [24]. It must be able to identify and correct any errors in sentence structure, such as run-on sentences, sentence fragments, or misplaced modifiers. It is notoriously difficult to evaluate these systems and demonstrate that they are safe to use. This is required because it must be able to identify and correct any errors in sentence structure, such as run-on sentences, sentence fragments, or misplaced modifiers.

All in all, automatic post-editing is a challenging task that requires a deep understanding of natural language processing, contextual understanding, sentence structure, style and tone, and accuracy [5]. It is a complex task that requires a sophisticated algorithm and a large amount of data to train the system.

Generally, datasets containing pairs work as the standard data source for most NLP problems. However, The training of APE models generally requires triplets including a source sentence (src), machine translation sentence (mt), post-edited output (pe), and reference sentence (ref/tgt). As considerable expert-level human labor is required in creating pe, APE researchers have encountered difficulty in constructing suitable datasets for most language pairs [25].

In our case, the source is in English (src_en), the machine-translated text we generate will be in Hindi (nmt_hi) and our reference/target Hindi sentence will be in Hindi as well (tgt_hi).

Triplet Corpus		
src_en (English Source)	**nmt_hi (LTRC-NMT Hindi Translation)**	**tgt_hi (Reference Hindi Translation)**
Browse the various methods of the current accessible	वर्तमान सुलभ के विभिन्न तरीकों को ब्राउज़ करें	इस समय जिसे प्राप्त किया गया हो, उसकी विभिन्न विधियों (मेथड) में विचरण करें

Table 1.0: Example of Source English, Hindi Translation and Post-Edited data triplet (as taken from the IIT-B parallel corpora.)

However, there is a considerable lack of data for this triplet. While there exists a vast vault of English-Hindi translation data, it is a bit harder to find the triplet that is required for APE. This is because, for each translated pair (English -> Hindi), we require a manually post-edited Hindi sentence - which will require multiple human annotators to create the said dataset. This in turn has resulted in the development of a specific problem statement that deals with increasing the size of annotated data by using synthetically generated data [26]. The objectives include understanding the State-of-the-Art in Automatic Post Editing and how these methodologies operate in Indian languages, specifically Hindi, and determining the impact of the size, type, and quality of the dataset on the performance of the APE models.

1.4 Research Questions

Given the vast nature of this field and its nascent existence, here are the specific research questions that we're trying to answer in this thesis:

1. How can one leverage the advances in deep learning and neural networks to improve the quality and efficiency of Automatic Post Editing in NLP, and what are the key technical challenges that need to be addressed in this regard?

2. What are the latest developments in the synthesis of artificial data? What effects does artificially generated data have on Neural APE systems? What is its effect on the quality of results?

3. How do the latest developments in the field of LLM and PLM models - alongside the advances in one-shot and few-shot learning - affect the problem statement of Automatic Post Editing?

Chapter 2

Evolution of Automatic Post Editing

In Chapter 2, we delve into the rich history and progressive evolution of Automatic Post Editing (APE), tracing its development from the early days of spell-checking and rule-based corrections to today's advanced, nuanced approaches that leverage statistical methods, neural networks, and domain-specific innovations. This exploration covers the inception of APE with tools like IBM's ”Memo” system, transitions through the phases of statistical [27] and rule-based methodologies [28] that addressed the limitations of machine translation outputs, and culminates in the cutting-edge neural methodologies, including the transformative impact of Transformers [29] and hybrid models. Special attention is given to the application of these technologies in enhancing the translation quality from English to Hindi, showcasing both the challenges encountered and the significant strides made in producing more accurate, contextually relevant translations. Through this journey, we examine how APE has continuously adapted to the evolving landscape of NLP, highlighting key academic contributions and the shift towards increasingly sophisticated models that promise greater efficiency and effectiveness in automatic text correction.

The field of APE has been studied for the past 50 years [18, 30]. Automatic post-editing, also known as automatic text correction or language processing, has a long history dating back to the mid-20th century. The field of Automatic Post-Editing in NLP continues to evolve rapidly, with new techniques and approaches being developed and refined on an ongoing basis. The increasing use of machine-generated text in various applications and the growing demand for accurate and high-quality translations make automatic post-editing an increasingly important area of research in NLP.

2.1 Spell Check and Rule-Based APE

One of the earliest efforts in this field was the development of spell-checking software. The first spell-checker called the ”Memo” system was developed by a team at IBM in the 1950s [31]. This was followed by the introduction of IMT - in which the target text under construction serves as the medium of communication between an MT system and its user [32]. Another well-known Automatic Post-Editing tool, Microsoft Word’s ”grammar checker,” was introduced in the early 1990s and was based on research by linguist Geoffrey Sampson [33]. The decade first started with Kevin et al. [34] writing about coping with article selection in Japanese to English translation by using rudimentary statistical methods. In the later parts of the 1990s, Machine Translation mostly followed a two-step methodology: a rule-based Machine Translation which is then followed by a compulsory human evaluation to correct the grammar [32]. These question problems were solved in a short time due to the unambiguous nature of the problem statement.

2.2 Statistical Methods of APE

Post this, Automatic Post-Editing has evolved into a field of not just spell checking but actually rephrasing and changing literature to make the content more relevant, suited, grammatically correct, and best fit for the context. The earliest studies were those that were developed before the maturation of deep learning. These initial studies explored the use of rule-based and statistical machine translation techniques for Automatic Post-Editing [35]. As early as 2005, Robert Frederking and Fei Xia continued work on the same lines by proposing a more detailed rule-based methodology for APE [17]. This was also the time where researchers started considering APE systems as being a necessary part of Machine Translation [36].

However, academia was quick to realize that Post Editing worked best when the context and domain were picked a-priori [37]. As pointed out by Parton et al. [38], domain-specific APE systems can improve MT output by exploiting information unavailable to the decoder, or by performing deeper text analysis that is too expensive at the decoding stage [39]. Since a majority of MT systems at the time were rule-based, it was guaranteed that any given input would give a fixed translated output every time the model was run. Due to this, rule-based and statistical APE methods worked to solve the problem with great efficiency [36].

By 2010, models by Nicola Bertoldi and Marcello Federico followed a statistical method for domain adaptation of statistical machine translation systems, which can be applied to APE [40]. The years 2011 and 2012 saw two versions of the DEPFIX system ([41], [42]) - a statistical rule-based APE for Czech translations. It attempts to correct some of the frequent

SMT system's errors in English-to-Czech translations by analyzing the target Czech sentence and using a morphological tagger and a dependency parser and attempts to correct it by applying several rules which enforce consistency with the Czech grammar [34].

Types of Statistical APE: A statistical method could be individually or a mix of:

- Pattern-Based Editing
- Rule-Based Post Editing

These approaches are based on the idea that languages share many common patterns and structures, which can be used to automatically correct errors in the text. These methods would employ techniques similar to those used by statistical machine translation models, such as Rule-Based Machine Translation (RBMT), to learn statistical rules specifically for post-editing, as shown in Figure 2.1. As illustrated, the source text is first processed through a rule-based machine translation system, which is then followed by a second rule-based system for automatic post-editing. These two models are often combined for enhanced performance.

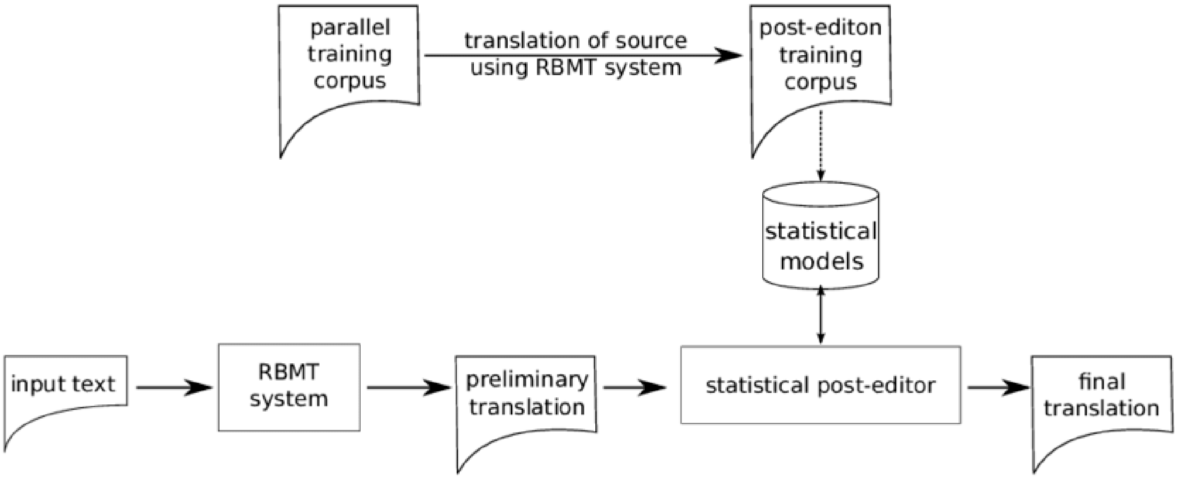

Figure 2.1: Architecture of a typical statistical post-editor [1]

However, this is also a case study on the limitations of rule-based methods where making the model (post-collection of data) required an extensive amount of manual work and even limitations on what dialects the model can work on [3]. The very same reasoning also extends to most work done in the space of Pattern-based APE models. Notable examples of this include the statistical approach proposed by Allen et al - ”Toward the development of a post-editing module for raw machine translation output: A controlled language perspective” (2000) [43], Isabelle et al. in ”Statistical Phrase-Based Post-Editing” [36], Dugast et al. in ”Statistical Post-Editing on SYSTRAN‘s Rule-Based Translation System” [44] and many more.

However, this field has not been deserted since the introduction of Neural methodologies. More recently, in the paper ”Automatic Post-Editing of English-Hindi Machine Translation Using a Rule-Based Approach” by Kaushik et al. (2019) [45], the authors propose a rule-based approach for APE in the context of English-to-Hindi machine translation. They use a set of pre-defined rules to correct common errors in machine-generated translations, such as word order and agreement errors. They evaluate their approach on a small test set and report promising results.

2.3 Neural Methodologies in APE

With the dawn of machine learning and neural nets, combined with the insurgence of sequential models like the RNN and Transformers [29], researchers started experimenting by combining the less time-and-data consuming statistical methods and the more accurate and powerful MT models.

More recently, with the development of deep learning techniques, existing neural machine translation (NMT) models have emerged as a popular approach to address Automatic Post-Editing. NMT uses deep neural networks to learn the relationships between words and their meanings and uses this knowledge to generate more accurate translations. In addition to these approaches, there has been growing interest in the use of reinforcement learning and other advanced machine learning techniques to further improve the accuracy and effectiveness of Automatic Post-Editing in NLP [46]. Overall, the field of Automatic Post-Editing in NLP continues to evolve rapidly, with new techniques and approaches being developed and refined on an ongoing basis.

Any top-of-the-line neural methodology for APE includes 2 main components. The first is a Sequence-to-Sequence model that acts similarly to an NMT model that takes in the translated sentence as input and aims to give an output similar to reference Hindi (refer to 3.5). It is noteworthy that all the sequential models referenced utilize one or more Transformers [29] which operate collaboratively within their architecture.

2.4 How are Neural models compared?

To assess the efficacy of new artificial intelligence models in improving automatic post-editing performance, it is essential to maintain consistency in the datasets used across all experiments. This approach ensures that any observed differences in performance metrics can be attributed solely to the model’s capabilities rather than variations in data. Consequently,

[0] https://pub.aimind.so/transformer-model-and-variants-of-transformer-chatgpt-3d423676e29c.

the same dataset is employed for all models to provide a controlled environment for accurate comparison [47].

The state-of-the-art in APE can be determined by comparing the quality metrics and scores of texts processed by different models. A key challenge in APE is evaluating the quality of the edited text. Several metrics have been developed to measure the quality of the output, such as TER [48], BLEU [49], and HTER [50].

2.5 APE models for Hindi

While most of the work we've discussed has been for languages such as English, there are also several academic papers that have explored this task for the Hindi language. Starting in 2016, Agarwal et al. first studied the effectiveness of different machine-learning techniques for APE in the context of English-to-Hindi machine translation [51]. They experiment with several models, including phrase-based and NMT models, and evaluate their performance on a small test set.

Throughout this period, in parallel, multiple studies focused on using hybrid methodologies that involved portions of statistical as well as neural techniques [52]. A good example of the same is the work done by Omkar et al. who devised a model that is the combination of phrase-based Statistical Machine Translation (SMT), example-based MT (EBMT), and rule-based MT (RBMT) [53]. Another example of this is a study from 2016, where Merve Nakir and Kemal Oflazer propose a hybrid approach that combines Statistical and Neural (similar to NMT) techniques for Automatic Post-Editing [54]. There also exists work that introduces custom neural/statistical models that are supposed to be language-agnostic [55] but need solid implementation with English-Hindi to confirm the claim. All these studies [56] demonstrate the growing interest in APE for the Hindi language and highlight the potential of different approaches for improving the quality of machine-generated translations. Further research in this area can help improve the accessibility of machine translation for Hindi speakers and facilitate communication across language barriers.

This work introduces novel contributions to Automatic Post-Editing (APE) for the Hindi language, addressing gaps in existing methodologies discussed in Chapter 2. Unlike previous studies focusing on widely studied languages, this research targets Hindi's unique morphological and syntactic complexities. In this study, we implement advanced neural methodologies, including Dual-Encoder Transformers, Transformer-to-Transformer models, and hybrid approaches combining neural networks with rule-based systems, tailored specifically for Hindi to enhance accuracy and contextual relevance. Additionally, innovative data augmentation techniques, such as forward and backward generation, are employed to expand training datasets, addressing the scarcity of triplet data. This comprehensive approach significantly advances APE for Hindi, setting a new benchmark for low-resource language translation quality.

.

Chapter 3

Experimental Insights: Automatic Post Editing

In this chapter, we present the methodology employed to enhance the quality of automatic post-editing (APE) systems. In this chapter, we explore the integration of neural network-based approaches, showcasing how advancements in deep learning have significantly improved APE model performance. The chapter further delves into the synthesis of artificial data, discussing its role in training robust APE systems and addressing both the challenges and benefits associated with synthetic datasets. In further chapters, we also analyze the impact of large language models (LLMs) and pre-trained language models (PLMs) on APE, with a focus on one-shot and few-shot learning methodologies. Key metrics and evaluation criteria for assessing the effectiveness of various APE models are outlined, establishing a framework for identifying state-of-the-art solutions. Finally, we provide a detailed description of our experimental setup, including the datasets, tools, and techniques used, to ensure the reproducibility and transparency of our research findings.

3.1 Overview of Experimental Setup

In the subsequent experiments, a consistent methodological approach was employed. All models were trained using the IIT-Bombay parallel corpora [10]. This corpora, developed by the Indian Institute of Technology Bombay, is a collection of bilingual text, specifically aligned at the sentence or phrase level. It encompasses over 1.5 million English-Hindi sentence pairs, sourced from 20 distinct origins. For clarity, in our upcoming experiments, the English sentence from each pair in the IIT-Bombay dataset will be denoted as **src_en** (source English), and its Hindi counterpart as **tgt_hi/tgt_hi** (target Hindi).

3.2 Machine Translation and Post-Editing Framework

The Machine Translation (MT) process is initiated by feeding src_en as the input and tgt_hi as the target, resulting in an output referred to as Machine Translated Hindi (nmt_hi). The output from this Post-Editing model is termed pe_hi, and a snippet of the final triplet dataset is exhibited in 3.1.

However, the nature of Automatic Post-Editing (APE) necessitates a triplet format. Consequently, aside from the original English-Hindi pairs, an additional element—the translated data—is integral. This data is derived from two sources: the LTRC Translation [57] and a custom transformer-based MT model. The LTRC translator, being the better Text-to-Text (T2T) model, is preferred for generating the new triplet, which comprises:

- **src_en**: English Source (English in the IIT-B pair)

- **nmt_hi**: Hindi output of the NMT model that took src_en as input

- **tgt_hi**: Reference Hindi Translation (Hindi in the IIT-B pair)

- **pe_hi**: Output of our APE model where src_en and nmt_hi are fed as parallel inputs.

Triplet Corpus		
src_en (English Source)	**nmt_hi (LTRC-NMT Hindi Translation)**	**tgt_hi (Reference Hindi Translation)**
Browse the various methods of the current accessible	वर्तमान सुलभ के विभिन्न तरीकों को ब्राउज़ करें	इस समय जिसे प्राप्त किया गया हो, उसकी विभिन्न विधियों (मेथड) में विचरण करें
Tests fundamental GUI application accessibility	मौलिक जीयूआई अनुप्रयोग पहुँच का परीक्षण करता है	मूलभूत जीयूआई अनुप्रयोग पहुंचनीयता का परीक्षण करता है
Browse the various methods of the current accessible	वर्तमान सुलभ के विभिन्न तरीकों को ब्राउज़ करें	इस समय जिसे प्राप्त किया गया हो, उसकी विभिन्न विधियों (मेथड) में विचरण करें

Table 3.1: Examples of APE Training triplet (as taken from the IIT-B parallel corpora and IIIT-H NMT system.)

3.3 Datasets and Model Selection

3.3.1 Overview

This study utilizes the IIT-Bombay (IIT-B) corpus [58], a comprehensive repository of bilingual text, which includes a distinct testing corpus with a considerable volume of unique English-Hindi sentence pairs. In addition to the primary corpus, supplementary datasets from external sources have been integrated to enrich the diversity and complexity of the test data. These augmentations are instrumental in evaluating the model's efficacy in replicating real-world translation scenarios and with heterogenus data. We will also discuss certain neural models that could be used to tackle this problem, pick what is proven in the industry to be

3.3.2 Datasets

This corpus encompasses over 1.5 million English-Hindi sentence pairs, meticulously annotated to ensure accuracy and relevance. The dataset is bifurcated into training and testing subsets, with the training set, referred to as HiEn-Train, comprising original sentence pairs, and the independent testing set, denoted as HiEn-Test, containing 2500 unique sentence pairs. Testing is performed over 3 test sets - the aformentioned HiEn-Test, FLORES200 with 997 pairs [59] and the third dataset being the NTREX-128 [60] corpus with 1997 pairs of its own. This approach provides a comprehensive evaluation of how an MT or APE model performs with both homogeneous and heterogeneous test datasets.

Train Data (After cleaning)		
Dataset	Source	Data Size (in Millions)
IIT Bombay English-Hindi Parallel Corpus	IIT-Bombay [58]	1.5

Table 3.2: Dataset Collection(s) used for Training

Test Data		
Dataset	Source	Size Data
IIT-B English-Hindi Parallel Test Corpus	IIT-Bombay	2500
FLORES200	Facebook Research	997
NTREX-128 [60]	Microsoft Research	1997

Table 3.3: Dataset Collections used for Testing

Given that the IIT-B corpus provides only an English source and its corresponding Hindi reference sentence per data point, it is necessary to generate machine-translated data to serve as an intermediary for our model as Automatic Post-Editing (APE) models require a machine-translated sentence as input and a parallel Hindi reference. Consequently, we must produce the machine-translated Hindi text by processing our English source through a translation system.

We experiment with two different models to perform this function and more will be discussed about the same later in this section.

3.3.3 Data Cleaning

In the initial phase of our data preparation, we commenced with a substantial dataset comprising 1,516,170 triplets. These triplets are critical for our study as they form the foundation upon which our machine-learning models are trained. The initial dataset, however, required a meticulous cleaning process to ensure the quality and relevance of the data used in the training process. This cleaning process involved several strategic steps aimed at refining the dataset to enhance the performance of our models.

The first step in our data-cleaning process was to address the issue of verbosity within the triplets. Specifically, we identified and removed any triplets in which the number of words for any component of the triplet exceeded 100 words. This threshold was established to eliminate excessively lengthy data points, which could potentially introduce noise and dilute the model's ability to learn from more concise, informative examples. By imposing this limit, we aimed to maintain a focus on data that was rich in information yet succinct enough to be effectively processed by our algorithms.

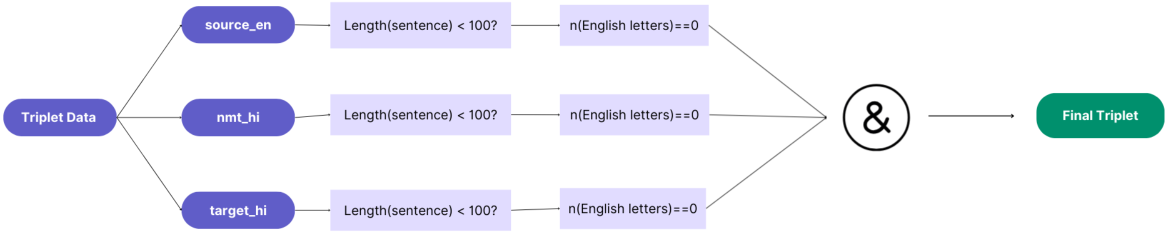

Figure 3.1: Flowchart detailing the data cleaning process.

Following the removal of verbose triplets, our next step targeted the presence of non-Latin scripts within our dataset. We specifically identified and removed any data from the Machine Translation (MT) or Post-Editing (PE) segments that contained script in a romanized form. This step was essential for maintaining consistency in the data format and ensuring that the training data was closely aligned with the input data expected during the model's deployment. It is advisable to either remove such foreign data or limit its use using a threshold for a number of foreign characters identified. The presence of romanized non-Latin scripts could lead to ambiguities and inconsistencies in the model's processing of textual information, thus adversely affecting its performance. A visual representation of this process can be seen in figure 3.1.

As a result of these cleaning operations, our dataset was refined to 1,506,696 triplets, a slight reduction from the original count but a necessary step towards ensuring the quality and consistency of our training data. It is important to note that these cleaning procedures were applied exclusively to our training dataset. We consciously chose not to apply these cleaning steps to the test or validation datasets. This decision was made to preserve the integrity and variability

of the test and validation data, ensuring that our model's performance could be accurately evaluated against real-world data that had not been preprocessed or altered.
To provide concrete examples of the types of data removed during our cleaning process, consider the following texts that were deleted (Note: All the text mentioned below is MT output):

- यह प्रोग्राम फ्री सॉफ्टवेयर है। आप इसे पुनर्वितरित कर सकते हैं और/या इसे gnu जनरल पब्लिक लाइसेंस की शर्तों के तहत संशोधित कर सकते हैं जैसा कि फ्री सॉफ्टवेयर फाउंडेशन द्वारा प्रकाशित किया गया है। लाइसेंस के किसी भी संस्करण 2, या (अपने विकल्प पर) किसी भी बाद के संस्करण. यह प्रोग्राम इस उम्मीद में वितरित किया जाता है कि यह उपयोगी होगा, लेकिन बिना किसी वारंटी के। यहां तक कि एक विशेष उद्देश्य के लिए व्यापारी या फिटनेस की निहित वारंटी के बिना. अधिक जानकारी के लिए जीएनयू जनरल पब्लिक लाइसेंस देखें। आपको इस प्रोग्राम के साथ-साथ जीएनयू जनरल पब्लिक लाइसेंस की कॉपी भी मिलनी चाहिए थी। अगर नहीं तो http://www. ग्नू। ओआरजी/लाइसेंस /

  In the sentence above, the substring "http://www." that is considered to be outside the scope of Devanāgarī script and hence was omitted.

- "पाठ प्रविष्टि में, अपनी नाव को नियंत्रित करने के लिए, प्रति पंक्ति एक कमांड दर्ज करें. समर्थित कमांड दो प्रवेश क्षेत्रों के बीच प्रदर्शित हैं. बाएं और दाएं कमांड डिग्री में कोण के द्वारा पीछा किया जाना चाहिए. कोण मान को बाईं या दाईं कमान के लिए एक पैरामीटर भी कहते हैं। डिफ़ॉल्ट रूप से 45 डिग्री का उपयोग किया जाता है। फॉरवर्ड कमांड एक दूरी पैरामीटर स्वीकार करता है. डिफ़ॉल्ट 1 द्वारा उपयोग किया जाता है. उदाहरण के लिए:-लेफ्ट 90:एक लंबवत लेफ्ट टर्न-फॉरवर्ड बनाएं 10:10 यूनिट के लिए आगे बढ़ें (जैसा कि शासक पर प्रदर्शित होता है)। लक्ष्य है स्क्रीन के अधिकार (लाल रेखा) तक पहुंचना। जब किया जाता है, तो आप अपने कार्यक्रम में सुधार करने की कोशिश कर सकते हैं और रेट्री बटन का उपयोग करके एक ही मौसम की स्थिति के साथ एक नई दौड़ शुरू कर सकते हैं. आप क्लिक करें और दूर और कोण में एक माप प्राप्त करने के लिए नक्शे पर कहीं भी अपने माउस खींच सकते हैं. अगले स्तर पर जाने से आपको अधिक जटिल मौसम की स्थिति मिलेगी।', "नावपर नियंत्रण रखने के लिये हर सीमापर कमांड दर्ज करे. एंटरी 'विभाग में उचित कमांड की सूची दिखाई है.' बांयी 'और 'दाहिनी' यह कमांड अंशात्मक कोणके साथ आती है. शुरूआत में अंशात्मक कोण की...."

The text above contains over 212 words, and since our filter removes all triplet pairs where any of the triplet pairs may have over 100 words, this was pruned out as well.

3.3.3.1 Observations

Here are some noteworthy observations made about the triplet data after it was cleaned with syntactic rules.

- 1/13th of the sentences with English text consisted of web url.

- Despite the dataset being touted as a parallel corpus with a single English sentence mapping to a single Hindi sentence, it has been found that there exist multiple data points may be a collection of 1-3 sentences mapped to one sentence/one group of Hindi sentences. Corpus segmentation doesn't seem to be uniform across the data.

- The machine-translated Hindi (mt_hi) and the post-edited Hindi (ape_hi) show variations in terminology and phrasing. For example, "Give your application an accessibility workout" is translated to "अपने एप्लीकेशन को एक्सेसिबिलिटी वर्कआउट दें" in mt and is post-edited to "अपने अनुप्रयोग को पहुंचनीयता व्यायाम का लाभ दें" in pe, indicating a refinement in language and possibly an attempt to use more natural or accurate Hindi expressions. This is further studied in the following chapters.

- Lexicon statistics: The average sentence length increases from the English source (src_en) with 13.42 words to the machine-translated Hindi (mt_hi) with 14.9 words, and further to the post-edited Hindi (tgt_hi) with 15.35 words. This suggests that translations tend to be longer than the original English sentences, possibly due to the nature of Hindi syntax or the addition of clarifying information. A cleaner look at this data can be seen in 3.6.

- The English source (src_en) has the largest vocabulary size with 186,738 unique words. This is expected as source texts typically encompass a wide range of topics and expressions. Interestingly, the vocabulary size decreases in the machine translation (mt_hi) to 131,474 and then increases again in the post-edited version (tgt_hi) to 157,086. The reduction in vocabulary size from src_en to mt_hi could indicate a limitation in the translation model's ability to capture the full diversity of the source language. The increase in tgt_hi suggests that post-editors are expected to reintroduce variety, possibly correcting or diversifying the language used in translations.

Apart from the changes listed above, there are no other changes being made to any of the datasets. It is to be noted once more that the test sets (IIT-B, FLORES, NTREX) are not being edited in any way whatsoever. The size of the triplet dataset before cleaning was 1,516,170 triplets. After the cleaning process, the dataset was reduced to 1,506,696 triplets. Cleaning the dataset resulted in a dataset reduction of approximately 9,474 triplets.

Table 3.4: File Statistics Comparison for Source (English) data

	Avg. Sentence Length	Median Sentence Length	Max Sentence Length	No. of unique words
Raw	13.42 words	7 words	679 words	433,538
Cleaned	11.92 words	8 words	99 words	276,845

Table 3.5: File Statistics Comparison for MT (Hindi) data

	Avg. Sentence Length	Median Sentence Length	Max Sentence Length	No. of unique words
Raw	13.46 words	8 words	1407 words	340,215
Cleaned	12.74 words	8 words	99 words	331,587

Table 3.6: File Statistics Comparison for Reference/Target (Hindi, tgt_hi) data

	Avg. Sentence Length	Median Sentence Length	Max Sentence Length	No. of unique words
Raw	13.47 words	8 words	1380 words	519,122
Cleaned	12.93 words	8 words	99 words	510,859

3.3.4 Data Architecture

The exploration of Automatic Post-editing (APE) in Hindi language translation highlights the shift from traditional Convolutional Neural Networks (CNN) [61] towards more sophisticated Sequence-to-Sequence (Seq2Seq) models [62], specifically focusing on Transformers. Initial attempts using CNNs revealed limitations due to data scale and bias, prompting a pivot to Transformer-based models for their superior handling of Seq2Seq challenges. Further studies went on to introduce novel Dual-Encoder Transformer models that incorporated dual encoders to process both the source English (src_en) and machine-translated Hindi (nmt_hi), leading to significantly improved translation accuracy [63] as evidenced by various metrics like BLEU score, CHRF, and TER across different datasets (IIT-B, FLORES200, NTREX). This approach underscores the importance of integrating source and translated inputs for enhancing post-editing efficiency.

3.4 Machine Translation

Since we need to first translate ENglish to Hindi to create our triplets, let's discuss Machine Translation. This section focuses on the selection of an optimal base translation model for Neural Machine Translation (NMT). To ascertain the most suitable model, a comparative analysis was conducted between the LocalNMT and LTRC NMT models, considering their performance metrics and training data characteristics.

But before we discuss the neural methodologies used in this study to convert English to Hindi text, it needs to be established that statistical methodologies dominated this field before the neural. While we've explored existing work in <APE and its different types - including Statistical APE methodologies - let's familiarise ourselves with the history of Statistical Machine Translation for Indian languages.

3.4.1 Statistical Machine Translation

Incorporating statistical methods into APE, particularly for Indian languages, utilizes data-driven models to predict and correct errors in machine-translated texts [64]. These methods analyze large corpora of monolingual and bilingual data to identify patterns and discrepancies in translation outputs [10]. This approach is crucial for Hindi and other Indian languages due to their linguistic complexity and diversity. Statistical models, like the Dual-Encoder Transformer, adapt to nuances in grammar [65], syntax, and semantics, facilitating accurate and contextually relevant translations. Adapting these models involves addressing challenges like script diversity and morphological richness, highlighting the importance of sophisticated statistical methods in refining APE processes.

Initial English-Hindi translation methods used rule-based approaches with linguistic rules and dictionaries, covering the entire MT process, from pre-processing [66, 67] to integral linguistic processes [68, 69]. Systems like AnglaHindi [70] combined statistics and example-based methods for more acceptable translations, especially in domain-specific cases [71]. Transfer-based MT systems, built on rule-based systems [72], bridge rule-based and statistical approaches by analyzing the source text, converting it to an intermediary representation, and generating the target text. These systems apply grammatical rules to parse the source language's structure, apply transformation rules, and synthesize the target text.

Hybrid methodologies combining statistical and rule-based MT have shown significant improvements in BLEU scores [73, 74]. Combining statistical MT, example-based MT, and rule-based MT potentially outperforms any subset [75], enhancing fluency, accuracy, and grammatical precision. These methodologies have been tested for Indian languages, including Bengali [76], Marathi [77], Telugu [78], and Malayalam [79]. Research continues to enhance both hybrid and individual methodologies—Statistical NMT and Rule-based NMT [80, 81, 82].

3.4.2 Neural Machine Translation

We will first start out by examining the different neural methodologies involved in the machine translation of English data to Hindi.

- **Convolutional Neural Network**

  Convolutional Neural Networks [83] (CNNs) have been pivotal in MT systems, especially before Transformers became widespread [84]. CNNs apply convolutional layers to input text, capturing local dependencies and patterns. Filters slide across the input data, extracting features at various levels of abstraction. Pooling layers then reduce dimensionality, retaining key features [85]. This makes CNNs suitable for tasks focusing on locality and invariance. However, CNNs struggled with long-range dependencies, impacting their effectiveness in complex sentence structures. These limitations led to the development of more practical models like RNNs [86].

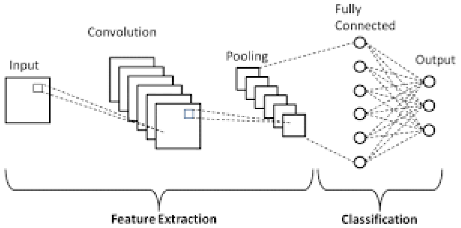

Figure 3.2: Different layers in a CNN model[1]

[1] https://www.upgrad.com/blog/basic-cnn-architecture/

- **RNN**

  RNNs excel in sequential learning due to their architecture, allowing the memory of previous inputs through self-connections [87]. This enables RNNs to process data sequences, making them superior for language modeling and MT [88]. They maintain a 'state' that helps in understanding context and sequence, outperforming CNNs in handling time-series tasks - including NMT [89, 90].

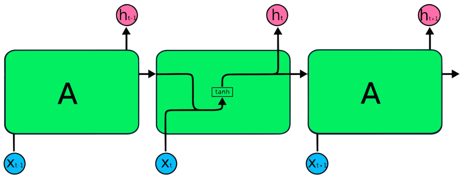

Figure 3.3: RNN architecture[2]

- **LSTM**

  RNNs have a key limitation: capturing long-term dependencies due to the vanishing gradient problem [91]. LSTMs, developed by Hochreiter & Schmidhuber in 1997 [92], address this by introducing gates that regulate information flow. These gates—input, forget, and output—enable LSTMs to maintain information over extended sequences effectively [92]. LSTMs significantly improve NMT systems by enhancing their ability to remember and utilize contextual information from the source language.

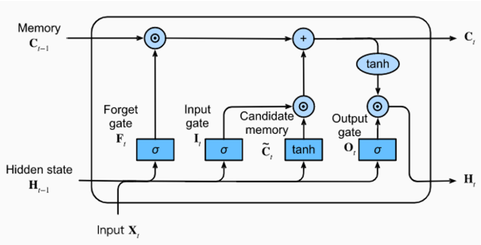
Memory $\mathbf{C}_{t-1}$
$\mathbf{C}_t$
tanh
Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Candidate memory $\tilde{\mathbf{C}}_t$
Output gate $\mathbf{O}_t$
σ
σ
tanh
σ
Hidden state $\mathbf{H}_{t-1}$
$\mathbf{H}_t$
Input $\mathbf{X}_t$

Figure 3.4: LSTM architecture[3]

[2] https://images.datacamp.com/image/upload/v1647442110/image6_f6vds6.png

[3] https://medium.com/@ottaviocalzone/an-intuitive-explanation-of-lstm-a035eb6ab42c

In the diagram 3.5, an English sentence is fed into LSTM cells within the encoder, and the aggregated representation is relayed to a decoder LSTM. This produces a machine-translated sentence. Although translations retain syntactic integrity, they may lack grammatical or semantic accuracy. Back-propagation, using the source/reference output as a guide, improves translation quality.

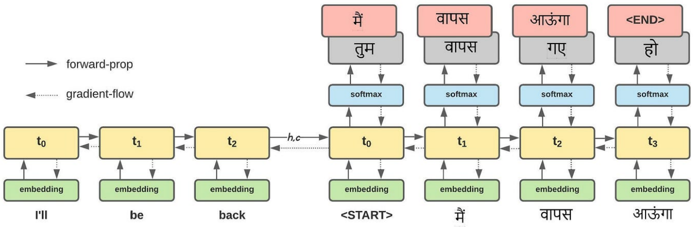

Figure 3.5: LSTM architecture[4]

- **Transformer**

LSTMs are limited due to their sequential processing nature, which prevents parallelization and have reduced efficiency when handling long sequences [29]. Transformers address LSTM limitations by enabling parallel sequence processing with self-attention mechanisms. This enhances efficiency and scalability in handling long sequences. Transformers model long-range dependencies and variable-length sequences effectively without sequential data processing. Examples include the Levenshtein Transformer [93] and the multi-source transformer by Tebbifakhr et al. [94]. These models show significant improvements over LSTMs in NMT tasks, leveraging contextual information from the source text [29].

[4] `https://towardsdatascience.com/english-to-hindi-neural-machine-translation-7cb3a426491f`

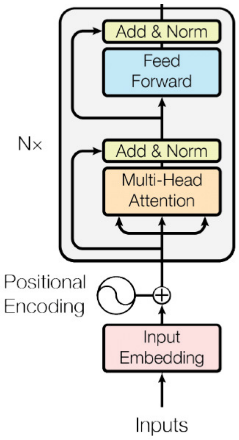

Figure 3.6: Single-Encoder Transformer[5]

Consequently, the architecture was modified to include a dual encoder that processes both nmt_hi and src_en data. This model begins with the dual encoder to generate a unified representation of both nmt_hi and src_en, which is then passed to a single Transformer-based decoder. The dual encoder processes parallel sentences from two languages to create a combined representation used to generate sequential output. This is visualized in 3.7 and later explored in depth.

[5] `https://wikidocs.net/167224/.`

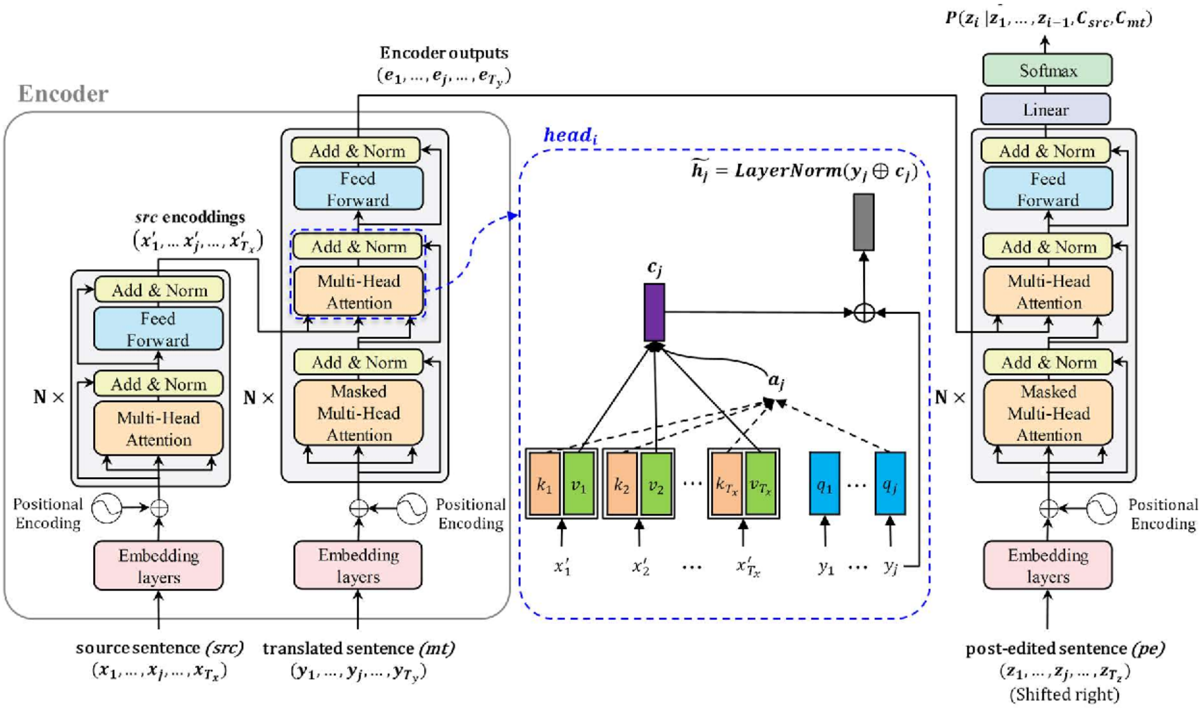
$P(z_i | z_1, \ldots, z_{i-1}, C_{src}, C_{mt})$
Encoder outputs
$(e_1, \ldots, e_j, \ldots, e_{T_y})$
Encoder
$head_i$
$\widetilde{h_j} = LayerNorm(y_j \oplus c_j)$
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
src encoddings
$(x'_1, \ldots x'_j, \ldots, x'_{T_x})$
c_j
a_j
k_1
v_1
k_2
v_2
k_{T_x}
v_{T_x}
q_1
q_j
x'_1
x'_2
x'_{T_x}
y_1
y_j
N ×
Positional Encoding
Embedding layers
source sentence *(src)*
$(x_1, \ldots, x_j, \ldots, x_{T_x})$
translated sentence *(mt)*
$(y_1, \ldots, y_j, \ldots, y_{T_y})$
post-edited sentence *(pe)*
$(z_1, \ldots, z_j, \ldots, z_{T_z})$
(Shifted right)

Figure 3.7: Dual-Encoder Transformer[6]

3.5 Comparing two NMT models

Before we analyze our model, let's define our evaluation metrics.

3.5.1 Evaluation Metrics for Automatic Post Editing of English-Hindi Translations

Evaluating the quality of automatic post-editing (APE) systems involves using several metrics that measure different aspects of translation quality. The following metrics are commonly used for this purpose:

3.5.1.1 BLEU (Bilingual Evaluation Understudy)

Description: BLEU [49] is a precision-based metric that evaluates the overlap of n-grams (contiguous sequences of words) between the machine-translated output and one or more ref-

[6] https://pub.aimind.so/transformer-model-and-variants-of-transformer-chatgpt-3d423676e29c.

erence translations. The score ranges from 0 to 1, where 1 indicates a perfect match with the reference translation.

Strengths for APE:

- *Widely Accepted:* BLEU is one of the most commonly used metrics in machine translation (MT) and is well-understood in the research community.
- *Precision Focused:* It effectively measures how many n-grams in the machine translation appear in the reference translation, providing a good indication of translation quality in terms of word overlap.

Limitations:

- *Lack of Recall:* BLEU does not account for recall, meaning it might not penalize missing translations.
- *Insensitive to Meaning:* It does not consider the semantic meaning or fluency, leading to potentially high scores for translations that are syntactically correct but semantically incorrect.

3.5.1.2 CHRF (Character n-gram F-score)

Description: CHRF [95] is based on the F-score of character n-grams. It evaluates the precision and recall of character n-grams of different lengths, making it more sensitive to small variations in word forms and better at handling inflected languages.

Strengths for APE:

- *Handles Morphological Variations:* Particularly useful for languages with rich morphology like Hindi, as it captures variations in word forms better than word-level metrics.
- *Balance of Precision and Recall:* By considering both precision and recall of character n-grams, CHRF provides a more balanced evaluation of translation quality.

Limitations:

- *Complexity:* Interpretation of the score can be less intuitive than word-level metrics.
- *Overweighting Small Changes:* Minor edits at the character level might overly influence the score, which may not reflect the overall translation quality.

3.5.1.3 TER (Translation Edit Rate)

Description: TER [48] measures the number of edits (insertions, deletions, substitutions, and shifts) required to change a system output into one of the references. It quantifies how much post-editing effort is needed to achieve a correct translation.

Strengths for APE:

- *Direct Measurement of Post-Editing Effort:* TER is particularly relevant for APE as it directly reflects the effort required to correct translations.
- *Interpretability:* The metric is straightforward and intuitive, as it directly counts the edits needed.

Limitations:

- *Simplicity:* While TER measures effort, it might oversimplify the nature of the edits, not distinguishing between minor and major changes.
- *Insensitivity to Fluency:* TER focuses on edit distance and may not fully capture improvements in fluency or readability.

3.5.1.4 COMET (Crosslingual Optimized Metric for Evaluation of Translation)

Description: COMET [96] is a neural network-based metric that uses pre-trained multilingual embeddings to evaluate translation quality. It compares the similarity of the embeddings of the machine-translated text and the reference translation, incorporating both source and target language information.

Strengths for APE:

- *Semantic Evaluation:* By leveraging neural embeddings, COMET captures semantic similarities and differences, providing a more meaningful evaluation of translation quality beyond surface-level word matching.
- *Contextual Understanding:* It evaluates translations in a more context-aware manner, making it suitable for assessing the improvements made by automatic post-editing systems.

Limitations:

- *Complexity and Computation:* COMET requires significant computational resources and understanding of neural network-based evaluations.
- *Less Established:* Being a relatively new metric, it might not be as widely adopted or understood as traditional metrics like BLEU.

3.5.1.5 Why These Metrics are Good for Automatic Post Editing

Automatic post-editing (APE) aims to correct errors in machine-translated text to improve its quality. Evaluating APE systems requires metrics that can accurately reflect the improvements made by post-editing.

Each metric brings unique strengths, offering a comprehensive evaluation framework that covers precision, recall, edit effort, and semantic quality, making them collectively well-suited for assessing automatic post-editing systems.

Let us now analyse the two NMT options evaluated for our study.

3.5.2 LTRC-NMT

The LTRC NMT model, designed for English-to-Hindi translation, uses a transformer architecture for machine translation, however, operates mostly as a ’black box’.

Developed by the International Institute of Information Technology, Hyderabad, this model utilizes a transformer-based sequence-to-sequence architecture, adept at processing English inputs and producing corresponding Hindi translations [57].

The translator makes use of the transformer model, known for its efficiency and accuracy in handling large sequences of data, is well-suited for the complexity of speech-to-speech translation.

The SSMT system is trained on extensive bilingual corpora from various Indian languages. The training data includes parallel sentences in multiple language pairs, such as English-Hindi, Hindi-Telugu, and Hindi-Gujarati. The dataset is collected from various sources, including publicly available text corpora, domain-specific texts, and contributions from native speakers to ensure high quality and diversity. It consists of over 25 Million pairs used for training.

3.5.3 LocalNMT

A custom model trained on a local setup with a GTX 1070 GPU, leveraging PyTorch and FairSeq [97] frameworks. This model primarily focuses on translating from Hindi to English, offering an insightful contrast to the LTRC-NMT's methodology. This model was trained on the cleaned IIT-B parallel corpora consisting of 1.5M pairs of English-Hindi parallel sentences. It's important to note the disparity in training data volume: the LocalNMT model was trained on approximately 1.5 Million English-Hindi pairs, while the LTRC-NMT had the advantage of over 25 Million pairs.

3.5.3.1 Performance Comparison

Since it has been well established that Transformers are superior to LSTMs which are superior to RNNs [29, 98, 99, 100, 101] for the task of machine translation, we stick with using the same to train the LocalNMT model. The performance of both models was rigorously assessed across three datasets: IIT-B, FLORES200, and NTREX. Key performance indicators include BLEU [49], CHRF [95], TER [102], and COMET [96]. The results, as tabulated below, indicate a consistently higher performance of the LTRC-NMT model over LocalNMT across all three datasets.

Scores for NMT output (IIT-B)				
Model	**BLEU**	**CHRF**	**TER**	**COMET**
LocalNMT	16.31	0.41	77.48	0.59
LTRC-NMT	21.59	0.46	69.93	0.63

Scores for NMT output (FLORES200)				
Model	**BLEU**	**CHRF**	**TER**	**COMET**
LocalNMT	19.71	0.43	71.10	0.61
LTRC-NMT	33.45	0.58	51.81	0.71

3.5.3.2 Results

There are two notable observations that we can make from the data in the tables.

Scores for NMT output (NTREX)				
Model	**BLEU**	**CHRF**	**TER**	**COMET**
LocalNMT	12.65	0.29	75.4	0.54
LTRC-NMT	25.69	0.51	60.0	0.61

Table 3.7: Comparing scores of the two machine translations models used in this study across 3 test datasets

- When comparing the two translation models, in the case of all three of the test datasets - IIT-B Test set, NTREX as well as FLORES200, the LTRC-NMT gives significantly better results than the LocalNMT. While both the NMT models are transformer-based, the local model - made using a simple transformer architecture and trained on the synthetic IIT-B dataset - is outdone by the transformer-based (but more complex) LTRC-NMT translation system. In fact, it beats LocalNMT even when the test set is chosen to be IIT-B test set.

- For the second observation, lets compare the difference between the scores of LTRC-NMT and LocalNMT on each dataset. The difference in BLEU scores for standard test datasets like FLORES200 and NTREX are 10+ in both cases, showing that when using a well researched organic open-source test dataset, the model is much superior. However, if we were to compare the scores of the models used on the IIT-B test dataset, the difference is marginal in comparison. This could be due to the IIT-B test set not being cleaned and also being synthetically generated/unreviewed by humans. Additionaly, LTRC-NMT appears to have a broader or more optimized training foundation, as evidenced by its superior performance on FLORES200 and NTREX datasets.

- The most significant observation emerges when comparing the performance across the three test sets. While the Automatic Post-Editing (APE) model indeed enhances the quality of the English-Hindi translations from the IIT-B train dataset, this improvement does not hold when the APE model is applied to heterogeneous datasets such as FLORES200 and NTREX. This indicates that while a basic NMT model, like LocalNMT, leaves substantial room for improvement through APE, the potential for quality enhancement diminishes when dealing with translations from a more sophisticated and efficient

NMT model. Thus, the effectiveness of APE is more pronounced with simpler NMT models, whereas high-quality NMT systems present a challenge for further post-editing improvements.

3.5.4 Dataset Specificity and Model Performance

The disparate nature of the FLORES200 and NTREX datasets plays a crucial role in influencing NMT model accuracy. FLORES200, encompassing general-purpose and common domain data, is particularly beneficial for research in low-resource languages. In contrast, NTREX focuses on specific domain texts, which impacts the NMT model's accuracy depending on the alignment between the training and testing data domains.

3.5.5 Conclusion and Future Implementation

Following this comparative analysis, it is evident that the LTRC NMT system outperforms the custom-built LocalNMT model. The effectiveness of the Automatic Post-Editing (APE) model in enhancing English-Hindi translations is more pronounced with simpler NMT models like LocalNMT, but diminishes with more sophisticated and efficient NMT systems, as evidenced by varied performance across the IIT-B, FLORES200, and NTREX test sets. Henceforth, the LTRC NMT system will be regarded as the gold standard for Neural Machine Translation. This benchmark will guide the implementation of Automatic Post Editing, leveraging the strengths of the LTRC NMT model for enhanced translation accuracy and efficiency. However, using the LTRC Model does not imply it is state-of-the-art across all existing models. This choice was based on convenience, stability, and the need to establish a baseline metric for comparison.

Chapter 4

Neural Architectures for Automatic Post Editing

This chapter explores the implementation of advanced neural architectures designed to enhance automatic post-editing (APE) in Hindi language translation. The chapter begins with an overview of the Transformer-to-Transformer (T2T) model, detailing its architecture and effectiveness in improving the quality of machine-translated text. This is followed by an examination of the Dual-Encoder Transformer (DET2T-PE) model, which integrates both source and machine-translated inputs to produce refined translations. The chapter presents empirical results demonstrating the superior performance of these models compared to baseline systems, underscoring the potential of neural methodologies in APE. The analysis includes a comparative assessment of these models against traditional approaches, highlighting significant improvements in translation accuracy and fluency. Overall, Chapter 4 provides a comprehensive evaluation of state-of-the-art neural models, showcasing their capabilities in addressing the complexities of Hindi APE and setting new benchmarks for translation quality.

4.1 Architecture of Automatic Post-editing in Hindi Language Translation

APE is inherently a Sequence-to-Sequence (Seq2Seq) challenge, dealing predominantly with monolingual data. In this context, our focus is on training the model using Hindi data (referred to as nmt_hi) to target the translated Hindi output (tgt_hi). This approach builds upon the premise that the initial translation phase is already completed, as detailed in the preceding chapter.

As Transformers work best for most tasks involving natural language due to their ability to remember or maintain a context as well as the benefits lent by the attention mechanism [29, 98, 99], we will begin experimentation with a simple single-encoder transformer model for APE.

4.1.1 Transformer-to-Transformer Model (T2T)

While we explored T2T as a neural architecture for NMT in the previous chapter, we will now define its use in our problem statement - along with the appropriate input and output. The Transformer-to-Transformer model architecture uses an adaptation of the Transformer architecture to capture semantic and syntactic properties of the input text.

Architecture: The architecture consists of 2 transformers - one for encoding and the other for decoding as shown in 4.1. In the case of Machine Translation, the encoder would first take the source language as an input and encode the same while the decoder would be focused on decoding the encoded representation by generating the appropriate target translation.

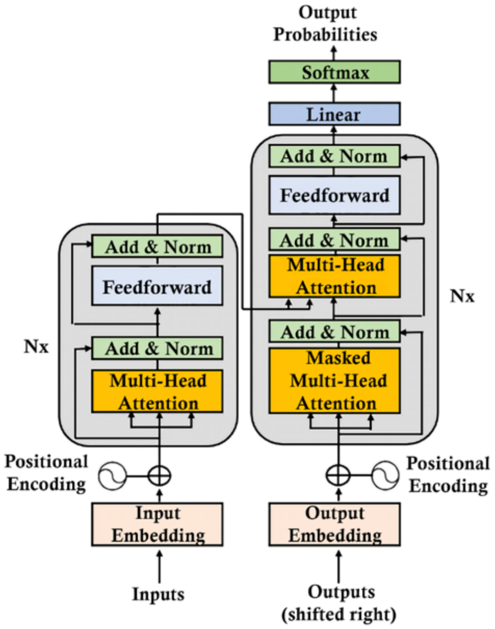

Figure 4.1: Simple Transformer-to-Transformer architecture that can be used in Machine Translation as well as Automatic Post Editing.[1]

Input and Output: In the case of APE, the input to the encoder involves the translated Hindi text (nmt_hi) while the input to the decoder/data for the decoder to train, we make use of reference Hindi Post-Edited output (tgt_hi). The aim is such that the output (tgt_hi) retains the meaning of the source text while correcting any errors present in the machine translation.

Experiment: The experiment was conducted with ~1.5M triplets and trained only on the IIT-B training set from the parallel corpora. The model was trained on a Google Colab notebook with access to a Nvidia A100 GPU with 40 GB of available VRAM. Of this, we consumed ~26 GB of VRAM and close to ~13 GB of RAM. The model was trained for exactly 25 spochs and took close to 26 hours.

The parameters as well as hyper-parameters for the encoder-decoder architecture (written in Fairseq) can be found in table 4.1.

[1] `https://pub.aimind.so/transformer-model-and-variants-of-transformer-chatgpt-3d423676e29c`

Result: The results for the use of the T2T model are outlined in table 4.3. While the metrics of this model are impressive, they make most sense when compared with the next suggested architecture. We shall return to discussing the results of this model later in the chapter.

Parameter	**Value**
`Encoder Architecture`	Transformer
`Decoder Architecture`	Transformer
`Dropout`	0.3
`Batch Size`	64
`Attention Dropout`	0.1
`Activation Dropout`	0.1
`Encoder Embed dimensions`	320
`Decoder Embed dimensions`	320
`Number of Encoder layers`	5
`Number of Decoder layers`	5
`Number of max epochs`	20
`Optimizer`	Adam
`Learning Rate`	0.0005
`Criterion`	Smoothed Cross Entropy
`Learning Rate Scheduler`	Inverse_sqrt

Table 4.1: Key Parameters for Fairseq Training Command

4.1.2 Dual-Encoder Transformer (DET2T-PE)

The model typically comprises two encoder components, each leveraging transformer-based neural networks to process and transform input sequences into high-dimensional vectors. These vectors are then compared, often via a dot product or a learned metric, to determine the inherent similarity or relationship between the two inputs. This approach proves particularly effective in tasks demanding an understanding of the inter-connectivity between two pieces of text or between text and image modalities, capitalizing on the transformer's strength in capturing contextual information within each sequence.

Architecture: This novel model architecture incorporates two parallel encoders [103], designated as DET2T-PE1 and DET2T-PE2. These encoders independently process their respective inputs, generating contextual representations. These representations are then fused and fed into a decoder to produce the post-edited output. However, the method to combine representations of the two encoders are explored in 2 ways in this study.

In a dual encoder transformer model for automatic post-editing (APE), there are four commonly used distinct methods for integrating information from source and machine-translated texts:

- **Concatenation:** This method combines the representations from both encoders by concatenating them along the feature dimension, which allows the decoder to access the complete information from both the source and the machine-translated text simultaneously. This method is straightforward but may not effectively highlight the interdependencies between the texts. [104].

- **Cross-attention:** This method leverages cross-attention mechanisms where the decoder attends to both the source and the machine-translated representations separately, allowing the decoder to dynamically integrate information from both encoders during decoding. Thus it potentially leads to more nuanced and contextually appropriate edits. [29].

- **Feature Fusion:** This technique merges the encoder representations using a learned linear transformation or a neural network to create a unified representation that incorporates features from both source and machine-translated text, enhancing the model's ability to generate accurate post-edits [105].

- **Cross-Modal Alignment:** This method aligns and combines information from different modalities (source and machine-translated text) using attention-based alignment techniques, enabling the model to effectively integrate and utilize the complementary information from both encoders [106].

While feature fusion and cross-modal alignment can be useful, they add complexity because they require advanced integration strategies and the alignment of different types of information

[107]. This complexity can make it harder to isolate the impact of each method on the post-editing results [108].

By focusing on concatenation and cross-attention, I aimed to provide a clear and focused evaluation without the extra variables introduced by more complex methods. This approach ensures that the results are easy to interpret and directly linked to the methods being tested, giving more precise insights into their strengths and weaknesses.

While concatenation merges information in a fixed manner, cross-attention offers a flexible, interaction-focused approach to combining textual representations. Literature also suggests that CrossAttention allows for dynamic interactions between the source and machine-translated texts, enhancing the model's ability to capture relevant context and dependencies, which is crucial for automatic post-editing[107]. Conversely, concatenation lacks this dynamic interplay, leading to less effective integration of information[108].

The parameters as well as hyper-parameters for the encoder-decoder architecture (written in Fairseq) can be found in table 4.2.

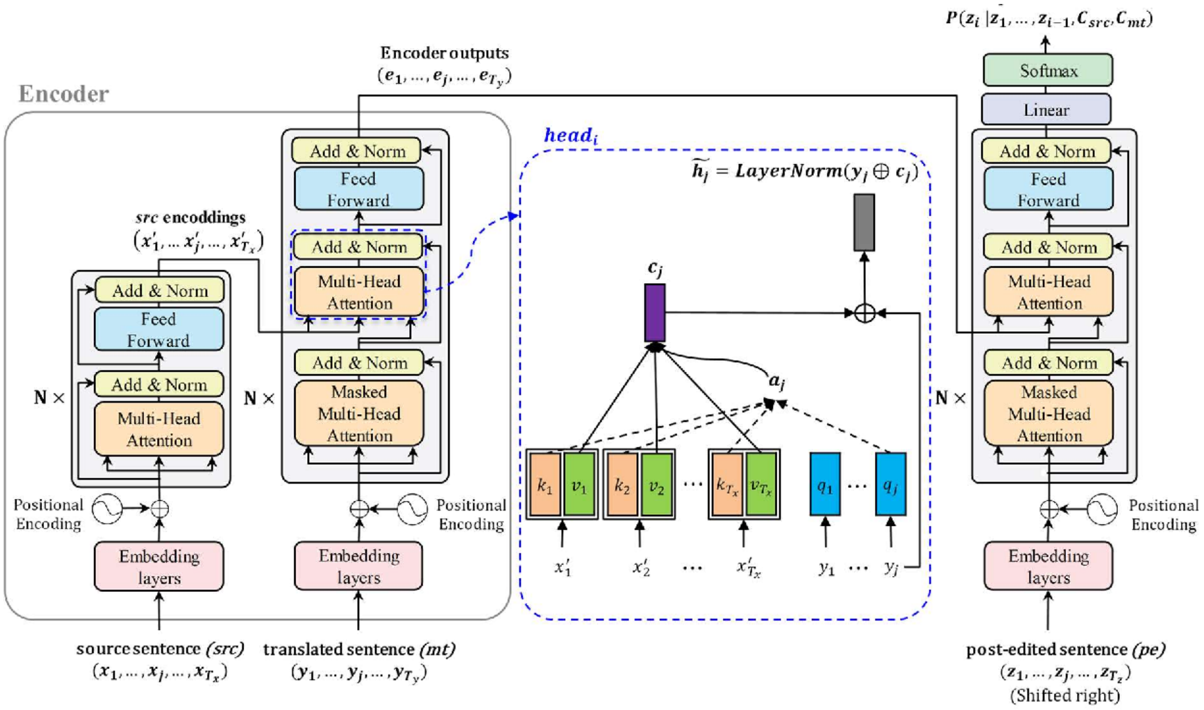
Encoder
Encoder outputs
$(e_1, \dots, e_j, \dots, e_{T_y})$
$P(z_i \mid z_1, \dots, z_{i-1}, C_{src}, C_{mt})$
Softmax
Linear
Add & Norm
Feed Forward
Multi-Head Attention
Masked Multi-Head Attention
src encoddings
$(x'_1, \dots x'_j, \dots, x'_{T_x})$
N ×
Positional Encoding
Embedding layers
$head_i$
$\widetilde{h_j} = LayerNorm(y_j \oplus c_j)$
c_j
a_j
k_1 v_1 k_2 v_2 ⋯ k_{T_x} v_{T_x} q_1 ⋯ q_j
x'_1 x'_2 ⋯ x'_{T_x} y_1 ⋯ y_j
source sentence (src)
$(x_1, \dots, x_j, \dots, x_{T_x})$
translated sentence (mt)
$(y_1, \dots, y_j, \dots, y_{T_y})$
post-edited sentence (pe)
$(z_1, \dots, z_j, \dots, z_{T_z})$
(Shifted right)

Figure 4.2: Dual Encoder model incorporating Source and Machine Translation as inputs, with tgt_hi as the target [2]

Input and Output: The two parallel encoders independently encode src_en (source English) and nmt_hi (machine-translated Hindi), respectively. Their outputs are then fed into a unified encoder, which combines DET2T-PE1 and DET2T-PE2, culminating in an amalgamated output (DET2T-PE3), which subsequently directs the neural network toward generating the final tgt_hi (target Hindi).

Experiment: The experiment was conducted with ~1.5M triplets and trained only on the IIT-B training set from the parallel corpora. The model was trained on a Google Colab notebook with access to a Nvidia A100 GPU with 40 GB of available VRAM. Of this, we consumed ~36 GB of VRAM and close to ~20 GB of RAM. The model was trained for exactly 25 spochs and took close to 38 hours.

Parameter	**Value**
`Source Language`	src
`Target Language`	pe
`Architecture`	Dual Encoder Transformer
`Dropout`	0.3
`Batch Size`	64
`Attention Dropout`	0.1
`Activation Dropout`	0.1
`Encoder1 Embed Dimensions`	320
`Encoder1 Layers`	5
`Encoder1 Attention Heads`	8
`Encoder2 Embed Dimensions`	320
`Encoder2 Layers`	5
`Encoder2 Attention Heads`	8
`Decoder Embed Dimensions`	320
`Decoder Layers`	5
`Decoder Attention Heads`	8
`Number of Max Epochs`	20
`Max Tokens`	3000
`Optimizer`	Adam
`Learning Rate`	0.0005
`Criterion`	Label Smoothed Cross Entropy
`Learning Rate Scheduler`	Inverse_sqrt
`FP16`	Enabled

Table 4.2: Key Parameters for New Fairseq Training Command

4.1.3 Results

The tables comparing the performance of Single Transformer (T2T) and Dual Encoder Transformer (DET2T-PE) models for Automatic Post Editing (APE) of English to Hindi translations across three datasets - IIT-B (4.3), FLORES200 (4.7), and NTREX (4.5) reveal insightful trends.

Scores for Neural Post-Editing models (IIT-B)						
APE Architecture	**Encoder Input**	**Decoder Output**	**BLEU score**	**CHRF**	**TER**	**COMET**
T2T	nmt_hi	tgt_hi	21.00	0.42	70.23	0.61
DET2T-PE (Concatenation)	src_en + nmt_hi	tgt_hi	11.01	0.21	83.71	0.53
DET2T-PE (Cross-Attention)	src_en + nmt_hi	tgt_hi	27.26	0.53	54.60	0.73

Table 4.3: Neural Post-Editing models performance on IIT-B dataset.

Comparing APE with NMT/Baseline results (IIT-B)				
Model	**BLEU score**	**CHRF**	**TER**	**COMET**
LTRC-NMT (NMT)	21.59	0.46	69.93	0.63
DET2T-PE (APE, Cross Attention)	27.26	0.53	54.60	0.73

Table 4.4: Comparing the NMT baseline to the best performing APE model (IIT-B).

Scores for Neural Post-Editing models (NTREX)						
APE Architecture	**Encoder Input**	**Decoder Output**	**BLEU score**	**CHRF**	**TER**	**COMET**
T2T	nmt_hi	tgt_hi	14.71	0.366	72.79	0.51
DET2T-PE (Concatenation)	src_en + nmt_hi	tgt_hi	6.57	0.27	89.12	0.42
DET2T-PE (Cross-Attention)	src_en + nmt_hi	tgt_hi	20.87	0.477	65.03	0.67

Table 4.5: Neural Post-Editing models performance on NTREX dataset.

Comparing APE with NMT/Baseline results (NTREX)				
Model	**BLEU score**	**CHRF**	**TER**	**COMET**
LTRC-NMT (NMT)	25.69	0.51	60.00	0.61
DET2T-PE (APE, Cross Attention)	20.87	0.47	65.03	0.61

Table 4.6: Comparing the NMT baseline to the best performing APE model (NTREX)

Scores for Neural Post-Editing models (FLORES200)						
APE Architecture	**Encoder Input**	**Decoder Output**	**BLEU score**	**CHRF**	**TER**	**COMET**
T2T	nmt_hi	tgt_hi	16.96	0.421	69.65	0.64
DET2T-PE (Concatenation)	src_en + nmt_hi	tgt_hi	7.21	0.21	86.94	0.57
DET2T-PE (Cross-Attention)	src_en + nmt_hi	tgt_hi	23.91	0.53	49.77	0.68

Table 4.7: Neural Post-Editing models performance on FLORES200 dataset.

Comparing APE with NMT/Baseline results (FLORES200)				
Model	**BLEU score**	**CHRF**	**TER**	**COMET**
LTRC-NMT (NMT)	33.45	0.58	0.52	0.71
DET2T-PE (APE, Cross Attention)	23.91	0.53	0.49	0.68

Table 4.8: Comparing the NMT baseline to the best performing APE model (FLORES200)

Notably, the DET2T-PE model with Cross-Attention consistently outperforms the other configurations across all metrics and datasets. For instance, on the IIT-B dataset, the Cross-Attention DET2T-PE model achieves a BLEU score of 27.26, a significant improvement over the T2T model's score of 21.00 and the Concatenation DET2T-PE model's 11.01. This pattern is echoed in the FLORES200 and NTREX datasets, where the Cross-Attention DET2T-PE model scores higher in BLEU and CHRF and lower in TER, indicating better translation quality and fewer errors.

The Concatenation DET2T-PE model consistently underperforms, suggesting that simply combining the source and machine-translated text representations is less effective than dynamically focusing on the relevant parts of each through Cross-Attention. This observation underscores the importance of sophisticated interaction between the dual encoders' outputs to capture the nuanced discrepancies between the machine-translated text and the source text, enabling more accurate and context-aware corrections.

These results demonstrate the efficacy of Cross-Attention mechanisms in leveraging the complementary strengths of dual encoders, significantly enhancing the quality of post-edited translations by providing a more nuanced understanding and integration of both the source and target texts. The consistent pattern across different datasets highlights the robustness and versatility of the Cross-Attention DET2T-PE model in improving APE outcomes.

4.1.4 Comparing performance with baseline

Another notable observation from the above is that while there is a 28% jump in BLEU scores for the IIT-B test set, where the post-edited output does score higher for all metrics, the same is not true for when comparisons are made of the results pertaining to the FLORES200 and NTREX datasets. So while APE seems to be improving quality of translation when tested with the test set that comes alongside the train dataset (which is also from the same IIT-B dataset), introducing a 3rd party dataset doesn't seem to improve the quality of machine translation.
This is in tandem with the results and observations found in other studies. Katriin in [109] proposes that while APE systems can better sentences that are poorly translated, they often fail to

beat the baseline for the majority. For the WMT'22 APE shared task [110] as well as WMT'23 APE shared task [111], almost 50% of the solutions using an architecture similar to DET2T-PE are able to beat the baseline results (plain machine translation, before any form of post-editing).

This only seems to reinforce the previously laid argument that domain and context significantly influence APE tasks. The heterogeneity and varying characteristics of different datasets can lead to divergent model performance, emphasizing the necessity of training APE models on diverse datasets to enhance generalizability or focusing on domain specific datasets only to train models used on text of that domain.

While discussed in detail in a later chapter, results from LLMs like ChatGPT - which are relatively dataset agnostic seem to perform equally well on all 3 test sets but again fail to show any significant improvement in terms of scores when compared to the baseline.

At this stage, we realize that the lack of sufficient training data is another major point of discussion that affects the efficacy of our model. The scarcity of resources necessitates the use of data augmentation techniques, such as forward and backward generation, to expand the dataset and improve model performance. This discussion transitions into Chapter 5, which explores advanced data augmentation methods to enhance model efficacy despite data limitations.

Chapter 5

Enhancing Model Efficacy through Data Augmentation

Chapter 5 delves into enhancing the efficacy of our automatic post-editing (APE) models through various data augmentation techniques. Given the scarcity of high-quality triplet data essential for training robust APE systems, especially for low-resource languages like Hindi, this chapter explores innovative methods to expand our training datasets. Initially, we discuss statistical models for data augmentation, such as random noising, which introduces controlled errors to simulate a more extensive dataset. We then transition to neural network-based augmentation techniques, including forward and backward generation, which leverage the power of neural models to generate synthetic data. The comprehensive analysis of these techniques demonstrates their impact on improving model performance. The chapter concludes with a detailed discussion of the results and implications of data augmentation.

5.1 Introduction to Data Augmentation

In the complex field of neural architectures, one effective approach to improve the quality of Post-Edited data is to keep the existing architecture but expand the training dataset. This can be done by generating synthetic data and adding it to the existing dataset. By expanding the training corpus in this way, we attempt to pad the dataset with additional data that has some resemblance to the original. This augmentation involves generating synthetic data, subsequently amalgamated into the existing dataset, thereby expanding the training corpus. Such expansion inherently elevates the likelihood of enhancing model performance [112]. Data augmentation can be achieved through two distinct methodologies: statistical and neural models. Our exploration begins with statistical models.

5.2 Statistical Models for Data Augmentation

- **Random Noising**

  In random noising [113], errors can be introduced via 3 operations: add, remove or replace. At any point - for any given sentence - only one operation (out of the 3) was carried out at any given point in time. Multiple operations can be performed on the same sentence, however, doing so would reduce the semantic quality of the sentence and likely not help increase the BLEU score for grammatically correct translated sentences. The first step here is to use a randomizer in order to pick one of the 3 aforementioned operations. For add and replace, we are required to also pick random words - which in turn requires us to tokenize all sentences in HPT and nmt_hi. Once this dataset of tokens is created, all stop words can then be removed from this list (adding stop words like ’a’, ’the’, ’an’ is not likely to change the semantics of a sentence for the better). For a similar reason, all terms with a frequency of greater than 5 have also been removed. If the operation chosen is ’deletion’, a random word is picked from the sentence and deleted. If the operation picked is ’addition’, a random token is chosen, after which a random index is chosen (where the index stands for which position this newly added word is supposed to fill, for example, an index of ’3’ would mean that we’re adding the randomly chosen word to become the 3rd word) from the newly generated sentence. If the operation chosen is ’replace’, firstly a random number is generated to decide the index of the word that has been chosen to be replaced. For example, if the random number chosen is 3, we’ll replace the 3rd word of the sentence with the new randomly chosen word.

  Once all these operations have been applied at random to each sentence in the corpus, the resultant dataset has over 3,000,000 data points/triplets. The next step would be to shuffle the entire dataset and prepare train and validation sets with an 80:20 split. Post this, the Dual-Encoder Transformer is trained from scratch on a combination of the old dataset and the newly generated dataset.

APE (IIT-B) + Random Noising							
Test Dataset	**Random Noising combined to training data?**	**APE Architecture**	**Test set size**	**BLEU**	**CHRF**	**TER**	**COMET**
IIT-B	Yes	DET2T-PE (Cross-Attention)	2507	28.13	0.59	51.90	0.66
IIT-B	No	DET2T-PE	2507	27.26	0.53	54.60	0.73
FLORES200	Yes	DET2T-PE (Cross-Attention)	997	17.01	0.47	68.22	0.52
FLORES200	No	DET2T-PE (Cross-Attention)	997	23.91	0.53	49.77	0.68
NTREX-128	Yes	DET2T-PE (Cross-Attention)	1997	14.17	0.33	69.91	0.56
NTREX-128	No	DET2T-PE (Cross-Attention)	1997	20.87	0.47	65.03	0.61

Table 5.1: Scores when training data has been supplemented with augmented data using random noising

The final results are conclusive. While every metric involving the IIT-B test set has improved with regards to using only the existing 1.5M pairs, with an improvement of around 5-10% for each metric, using a model trained on Random Noising does not do anything to improve the performance of the model on FLORES200 and NTREX-128 scores. The very same reasoning given in the past - about the homogeneity of test and train data - can be applied to explain the trends captured in this study as well.

- **MLM Noising**

  MLM Noising is an advanced form of Random Noising that uses parallel corpora to generate synthetic machine translations (MT) with controlled error distributions similar to standard MT outputs. This method involves tokenizing sentences, assigning part-of-speech (POS) tags, and constructing a wordnet based on frequency and adjacency.

  The process includes analyzing frequently altered POS tags, creating a confusion matrix to understand these changes, and developing a dictionary from all tokens (excluding stop words and high-frequency terms). Sentences in nmt_hi are then represented as lists of tokens with their respective POS tags. Tokens are selected based on probability, and their addition direction (before or after the existing token) is randomized. The token is matched with a POS tag from the confusion matrix and a corresponding word from the dictionary is selected for inclusion or replacement.

  For example, editing probabilities based on POS tags might be:

  - 54% of verbs are replaced.
  - 30% of nouns are added.
  - 16% of verbs are transformed into adjectives.

  While MLM Noising is effective, given its heavy overhead for implementation, it was not employed or replicated in this study as the focus was more on neural methodologies rather than statistical approaches.

5.3 Neural Network-Based Data Augmentation

Following exploring statistical models, the focus shifts to neural network-based data augmentation methods.

5.3.1 Forward Generation

Forward Generation hinges on the premise that existing nmt_hi data closely approximates the target Hindi sentence (tgt_hi) [114]. For each src_en-nmt_hi pair, the model uses src_en and nmt_hi as inputs to generate the Automatic Post Edited output of the model - with the target being tgt_hi and the actual model output being referred to as pe_hi. Subsequently, for each src_en-nmt_hi pair, a corresponding src_en-pe_hi pair is created, effectively doubling the dataset to approximately 3M pairs[112].

During training, the APE model is provided with both src_en-nmt_hi and src_en-pe_hi pairs, with tgt_hi as the target output. These additional 1.5M synthetic triplets are designated as FWG (Forward-Generated).

In the figure below, <—> represents the output of the APE/MT Model on the left hand side and the target being used to train on the right-hand side. The new datasets being created in the figure 5.1 are (src_en, nmt_hi) concatenated with (src_en, pe_hi) for encoder input and tgt_hi being the appropriate reference.

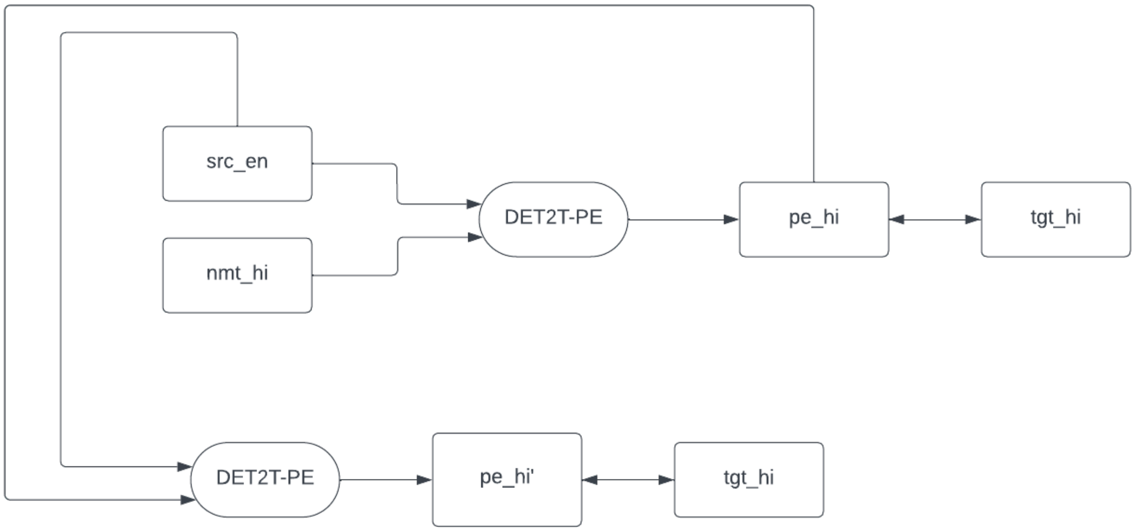

Figure 5.1: Forward Generation being used to create a synthetic dataset which is combined with the original dataset

Experiments with Synthetically Generated Data (IIT-B)						
Training Data	**APE Architecture**	**Train Dataset Size (Number of Triplets)**	**BLEU score**	**CHRF**	**TER**	**COMET**
HiEn-Train + FWG	DET2T-PE (Cross-Attention)	3013392	29.23	0.55	0.53	0.55

Table 5.2: Scores when training data is increased in size by using data augmented using Forward Generation

5.3.2 Backward Generation

Backward Generation capitalizes on the idea of introducing errors into tgt_hi by learning from nmt_hi patterns. Initially, nmt_hi is produced from src_en using a preferred MT model. The src_en-tgt_hi pair is then fed as input into the dual encoder with the generated nmt_hi as the target. This process facilitates the model in learning the necessary modifications to convert tgt_hi data into nmt_hi format [114].

This technique also doubles the dataset size. The output of this process, labeled HSPE (Hindi Simulated Post-Edited English), is utilized in place of nmt_hi in new training triplets, creating a unique set of synthetic data referred to as BWG (Backward-Generated) [112]. In figure 5.2, the new triplet created is (src_en, HSPE, tgt_hi) which is then concatenated with the previously existing (src_en, nmt_hi, tgt_hi)

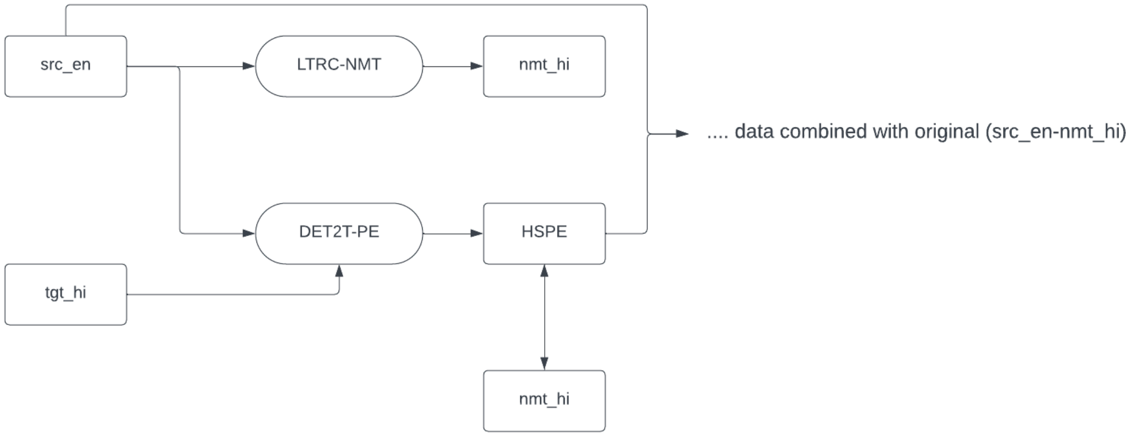

Figure 5.2: Backward Generation being used to create a synthetic dataset which is combined with the original dataset

Experiments with Synthetically Generated Data (IIT-B)						
Training Data	**APE Architecture**	**Train Dataset Size (Number of Triplets)**	**BLEU score**	**CHRF**	**TER**	**COMET**
HiEn-Train + BWG	DET2T-PE (Cross-Attention)	3013392	27.59	0.55	0.53	0.52

Table 5.3: Scores when training data is increased in size by using data augmented using Forward Generation

5.3.3 Backward Translation

Backward Translation addresses data augmentation by altering the initial stage of the translation process, src_en. Here, src_en is translated into nmt_hi, which is then back-translated into English, termed 'mt_en'. This process results in a version of the text that has undergone two translation cycles.

For each src_en-nmt_hi-tgt_hi triplet, a new mt_en-nmt_hi-tgt_hi triplet is generated, contributing an additional 1.5M synthetic triplets, collectively known as BWT (Backward-Translated) [114].

Backward translation - in both studies Hindi as well as Marathi (as will be presented later) - returns significantly detrimental results and hence can be seen as worsening performance, even when used alongside other methodologies that help improve performance. For this purpose, we will leave out Backward Translation from the final train dataset [47]. Below, you can see the results obtained when data augmented using BWT is included in the final train dataset. The new triplet being generated is (mt_en, nmt_hi, tgt_hi) which is concatenated with (src_en, nmt_hi, tgt_hi).

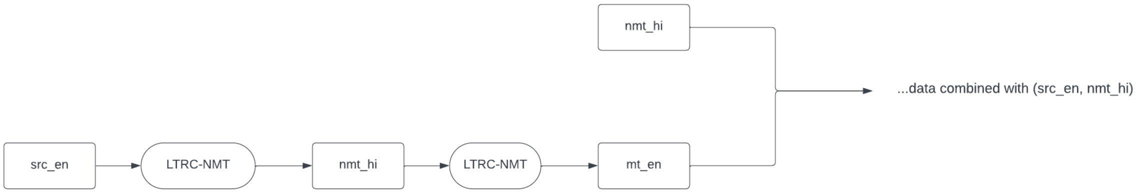

Figure 5.3: Backward Translation being used to create a synthetic dataset which is combined with the original dataset

Experiments with Synthetically Generated Data (IIT-B)						
Training Data	**APE Architecture**	**Train Dataset Size (Number of Triplets)**	**BLEU score**	**CHRF**	**TER**	**COMET**
HiEn-Train + BWT	DET2T-PE (Cross-Attention)	3013392	10.97	0.33	0.63	0.33

Table 5.4: Scores when training data is increased in size by using data augmented using Backward Translation

5.3.4 Final Result

Combining all three well-performing datasets - HiEn-Train (IIT-B Train data), FWG (Forward Generation Augmented Data), and BWG (Backward Generation Augmented Data). We are excluding the BWT (Backward Translation Generated) dataset as it only decreases the efficacy of the translation+APE model.

Experiments with Synthetically Generated Data (IIT-B)						
Training Data	**Architecture**	**Train Dataset Size (Number of Triplets)**	**BLEU score**	**CHRF**	**TER**	**COMET**
HiEn-Train	DET2T-PE (Cross-Attention)	1,500,000	27.26	0.53	0.55	0.73
HiEn-Train + Random Noising	DET2T-PE (Cross-Attention)	3,013,392	28.13	0.59	0.52	0.66
HiEn-Train + FWG	DET2T-PE (Cross-Attention)	3,013,392	29.23	0.55	0.53	0.55
HiEn-Train + BWG	DET2T-PE (Cross-Attention)	3,013,392	27.59	0.55	0.53	0.52

HiEn-Train + FWG + BWG	DET2T-PE (Cross-Attention)	4,520,088	26.91	0.47	0.76	0.50

Table 5.5: Comparing scores where training data was augmented using different neural methadologies

5.4 Results and Discussion

The integration of the synthetically generated data, amounting to 1.5M triplets per method (comprising forward generation and backward generation), with the original dataset of 1.5M triplets, has been instrumental in expanding the training corpus. The impact of this augmentation on the model's performance is presented in the following section, highlighting the efficacy of each methodology in enhancing computational linguistic applications. While Forward Generation adds a decent score increase to the original dataset, backward generation seems to hover more or less at a similar score as experiments that worked with HiEn-Train.

The table presents a comparison of the performance of an Automatic Post Editing (APE) model under different training scenarios, utilizing synthetic data generation techniques to augment the training dataset. The methodologies examined include Forward Generation (FWG), Backward Generation (BWG), and Backward Translation (BWT), with the performance metrics being BLEU score, CHRF, and TER.

A notable observation is a substantial improvement in BLEU score when FWG is employed alongside the base HiEn-Train dataset, indicating a significant enhancement in the model's ability to generate text closely aligning with the reference translations. This method not only increases the dataset size but also seems to enrich the model's training with diverse linguistic constructions, leading to more accurate translations as evidenced by a BLEU score increase from 27.26 to 29.23, alongside improvements in CHRF and a reduction in TER.

Conversely, the addition of BWG alone does not appear to significantly alter the model's performance in terms of BLEU score, suggesting that backward-generated texts might not provide the same level of beneficial diversity or complexity as forward-generated ones. This is further evidenced by the slight decrease in BLEU score compared to the base dataset. Interestingly, the use of BWT dramatically reduces the BLEU score, highlighting a potential mismatch between the back-translated texts and the model's target output. This could indicate that the nuances introduced by back translation do not align well with the model's learning objectives or possibly introduce noise rather than useful variance.

Combining the original dataset and two of the three discussed methodologies (FWG, BWG, IIT-B train) results in a dataset that, while still substantial in size, yields a lower BLEU score than the FWG-augmented set alone. This composite approach seems to dilute the positive impact seen with FWG, possibly due to the conflicting or less coherent linguistic patterns introduced by BWG. The increase in TER and decrease in CHRF compared to the base dataset and FWG-augmented set further support this, suggesting that while augmentation can significantly benefit APE models, the type of augmentation and its integration into the training process need careful consideration to avoid diminishing returns or adverse effects on model performance. The architecture used in all the above experiments was DET2T-PE (Cross-Attention).

5.4.1 Reduction in COMET Scores

Despite the promising scores of the BLEU, CHRF, and TER metrics showing improvement or sustained levels of quality, in all three cases, the calculation of the COMET score has shown a significant decrease. In fact, every augmentation technique shows a decrease of 25-28% when compared to COMET scores of experiments that did not utilize any neural data augmentation methods. COMET is a neural-based evaluation metric that considers semantic and contextual accuracy. The substantial drop in COMET scores for augmented data experiments indicates that while FWG and BWG improve surface-level matching, they may introduce noise affecting deeper semantic alignment. Synthetic data might not capture nuanced meanings and contextual appropriateness as effectively as real data, leading to translations that are lexically close but semantically misaligned with the reference.

At the same time, this does not necessarily mean that the new post-edited data is particularly bad at its job. A myriad of reasons - right from the quality of data to the purpose of the experiment could weigh in to create this effect. Later on in this chapter, we manually study and annotate a large chunk of this Post-Edited data and several observations from the same can explain this trend of COMET scores.

5.4.2 Strategies to Overcome the Discrepancy

Quality Control for Synthetic Data: Implementing stricter quality checks on generated synthetic data can ensure high semantic fidelity. Advanced models and filtering techniques can help maintain the quality of synthetic sentences.

Fine-Tuning with Real Data: After initial training on augmented data, a fine-tuning phase using high-quality real data can align translations more closely with human-like semantic understanding, improving COMET scores.

Hybrid Approaches: Using a balanced combination of multiple data augmentation techniques and real data in a hybrid training approach can harness the benefits of augmentation while mitigating noise introduction.

Therefore, it is crucial to consider multiple evaluation metrics and adopt a balanced approach to data augmentation and validation to achieve comprehensive improvements in translation quality.

5.4.3 Conclusion

In summary, data augmentation methods are essential for improving the performance of APE models. However, the trade-off between surface-level accuracy and deep semantic understanding necessitates a careful and balanced approach. By implementing quality control measures, fine-tuning with real data, and adopting hybrid training approaches, it is possible to achieve balanced improvements across all evaluation metrics.

5.5 Manual Analysis of APE Data Generated

While evaluation metrics provide a clear picture of the differences in quality between Machine Translated (MT) and Automatic Post Edited (APE) data, a deeper understanding was sought through manual analysis. We manually annotated and compared 100 triplets (Source English - Machine Translated Hindi - Automatic Post Edited Hindi) in accordance with Multidimensional Quality Metrics (MQM) [115]. Each of the 24 different types of errors was assigned a unique ID for reference.

A random sample of 100 triplets was taken from the IIT-B test set, along with the corresponding APE sentence. Each triplet consisted of the English source ('src' from the IIT-B dataset), the Machine Translated Hindi sentence ('mt' generated using the IIIT-H translation model), the reference Hindi sentence ('ref' from the IIT-B test set), and the APE Hindi sentence ('ape' generated by passing 'src' and 'mt' through our best-performing locally trained APE model, HiEn-Train + FWG).

For each record, we studied mt_hi and pe_hi individually and annotated them with errors we found in the same. In order to be fair and utilize the privilege of not having to stick to a pre-determined target string for translation, we analyzed these manually and in detail, but not in comparison with tgt_hin. By studying these comparisons and identifying the different MQM tags [115] in each, we aimed to understand:

1. How the machine-translated data compares to the reference Hindi sentence. This analysis helps identify the strengths and weaknesses of our machine translation model, which can be particularly insightful given the model's "black box" nature.

2. How Automatic Post Editing modifies the machine-translated sentence. This study examines whether and how APE improves or diminishes the quality of the translated sentence.

3. Whether there are discernible patterns in the APE model's corrections, or if the post-editing process is consistent across different sentences.

So, MT as well as APE were both studied and assigned tags corresponding to the appropriate MQM error. Alongside the error ID and the text associated with it, each tag was also assigned a severity score between 0 and 3, in ascending order of severity of the error. So an error with severity 3 would be considered a major error and more disruptive than an error with a severity of 2. A severity of 1 would attribute to an error that can be overlooked without any loss in translation quality.

ID	Type	Subtype	Explained
1	Terminology	Inconsistent with terminology resource	Use of a term that differs from term usage required by a specified termbase or other resource.
2	Terminology	Inconsistent use of terminology	Use of multiple terms for the same concept in cases where consistency is desirable.
3	Terminology	Inconsistent case of terminology	Use of multiple terms for the same concept in cases where consistency is desirable.
4	Terminology	Wrong term	Use of term that it is not the term a domain expert would use or because it gives rise to a conceptual mismatch.
5	Accuracy	Mistranslation	Error occurring when the target content does not accurately represent the source content.
6	Accuracy	Overtranslation	Error occurring in the target content that is inappropriately more specific than the source content.
7	Accuracy	Undertranslation	Error occurring in the target content that is inappropriately less specific than the source content.
8	Accuracy	Addition	Error occurring in the target content that includes content not present in the source.
9	Accuracy	Omission	Error where content present in the source is missing in the target.
10	Accuracy	Do not translate	Error occurs when a text segment marked ”Do not translate!” is translated in the target text.

11	Accuracy	Untranslated	Error occurring when a text segment that was intended for translation is omitted in the target content.
12	Linguistics Conventions	Grammar	Error that occurs when a text string (sentence, phrase, other) in the translation violates the grammatical rules of the target language.
13	Linguistics Conventions	Punctuation	Punctuation incorrect according to target language conventions.
14	Linguistics Conventions	Spelling	Error occurring when a word is misspelled.
15	Linguistics Conventions	Unintelligible	Text garbled or incomprehensible.
16	Linguistics Conventions	Character encoding	Error occurring when characters garbled due to incorrect application of an encoding.
17	Linguistics Conventions	Textual Conventions	Error that occurs when a text string (word, phrase, sentence, phrase, other) violates the text-building (discourse) norms of the target language.
18	Style	Organisational Style	Errors occurring where text violates third-party style guidelines.
19	Style	Third party style	Errors occurring where text violates third-party style guidelines.
20	Style	Inconsistent with external reference	Errors occurring when text fails to conform with a declared external style reference.

21	Style	Language Register	Characteristic of text that uses a level of formality higher or lower than required by the specifications or general language conventions.
22	Style	Awkward Style	Style involving excessive wordiness or overly embedded clauses, often due to inappropriate retention of source text style in the target text.
23	Style	Unidiomatic style	Style that is grammatical, but unnatural.
24	Style	Inconsistent Style	Style that varies inconsistently throughout the text.

5.5.1 Detailed Analysis of Three Triplets

Example 1

Triplet:

- **Source:** ”As a result, the seatbacks fail to comply with federal auto safety standards on head restraints.”
- **MT:** "नतीजतन, सीटबैक सर पर संयम से संघीय ऑटो सुरक्षा मानक का पालन करने में विफल रहता है।"
- **APE:** "परिणामस्वरूप सीटबैक, हेड रेस्ट के बारे में संघीय ऑटो सुरक्षा मानक का अनुपालन करने में विफल रहा।"

Errors and Tags:

- **Terminology: Wrong Term (ID 4)**
  - **MT Error:** "सर पर संयम" for ”head restraints.”
  - **APE Error:** "हेड रेस्ट" is a direct transliteration, not an appropriate translation.
- **Accuracy: Mistranslation (ID 5)**
  - **MT Error:** "सर पर संयम" does not accurately represent ”head restraints.”
  - **APE Error:** While it corrected the term, it introduced a transliteration issue.
- **Accuracy: Untranslated (ID 11)**
  - **MT Error:** ”seatbacks” is transliterated to "सीटबैक."
  - **APE Improvement:** No such error present.
- **Style: Unidiomatic Style (ID 23)**
  - **APE Error:** Use of "हेड रेस्ट" remains unidiomatic.

Conclusion: MT had significant terminology and mistranslation issues. APE improved accuracy but introduced an unidiomatic style, showing a need for better contextual adaptations.

Example 2

Triplet:

- **Source:** ”Director, Dr. S.K. Garg, said that the institution’s annual function is celebrated as the foundation day.”
- **MT:** "निदेशक डॉ. एस. के. गर्ग ने कहा कि संस्थान का वार्षिक समारोह स्थापना दिवस के रूप में मनाया जाता है।"
- **APE:** "निदेशक डॉ. एस.के. गर्ग ने कहा कि संस्थान का वार्षिकोत्सव स्थापना दिवस के रूप में मनाया जाता है।"

Errors and Tags:

- **Terminology: Inconsistent Use of Terminology (ID 2)**
  - **APE Error:** Changed "वार्षिक समारोह" to "वार्षिकोत्सव," causing inconsistency.

Conclusion: MT was accurate, but APE introduced a terminology inconsistency. The focus should be on maintaining consistent terminology in APE.

Example 3

Triplet:

- **Source:** ”At around 2.15pm on Wednesday, an eagle-eyed dog walker spotted Ruby on the ledge in the quarry, stranded 50ft up.”
- **MT:** "बुधवार को लगभग 2.15 बजे, एक ईगल-आइड डॉग वॉकर ने खदान में लेज पर रूबी को देखा, 50 फुट ऊपर फंसी।"
- **APE:** "बुधवार को दोपहर 2.15 बजे, एक तेज़ नज़र वाले डॉग वॉकर ने रूबी को, जमीन से 50 फुट की ऊंचाई पर खदान में पहाड़ पर देखा।"

Errors and Tags:

- **Terminology: Wrong Term (ID 4)**
  - **MT Error:** "ईगल-आइड" is a transliteration.
  - **APE Improvement:** Corrected to "तेज़ नज़र वाले."
- **Accuracy: Mistranslation (ID 5)**
  - **MT Error:** "लेज पर" does not convey the full sense of danger.
  - **APE Error:** "पहाड़" exaggerates the scenario.
- **Style: Unidiomatic Style (ID 23)**
  - **APE Error:** "डॉग वॉकर" is unidiomatic.

Conclusion:

- The analysis shows that while APE can correct certain types of errors, it may introduce new ones, especially in the style category.

Conclusion: MT had terminology and mistranslation issues. APE corrected terminology but introduced exaggeration and unidiomatic style, indicating a need for balancing accuracy and idiomatic expression.

5.5.2 Results

Having done qualitative analysis by manually studying each triplet, some quantitative evaluation can also shine a light on different aspects of the study.

For example, when counting the number of individual errors across the entire study (each sentence with multiple errors will count as multiple errors and not just one), we observe that the difference between the number of errors in MT and APE is relatively small, with 124 and 96 respectively.

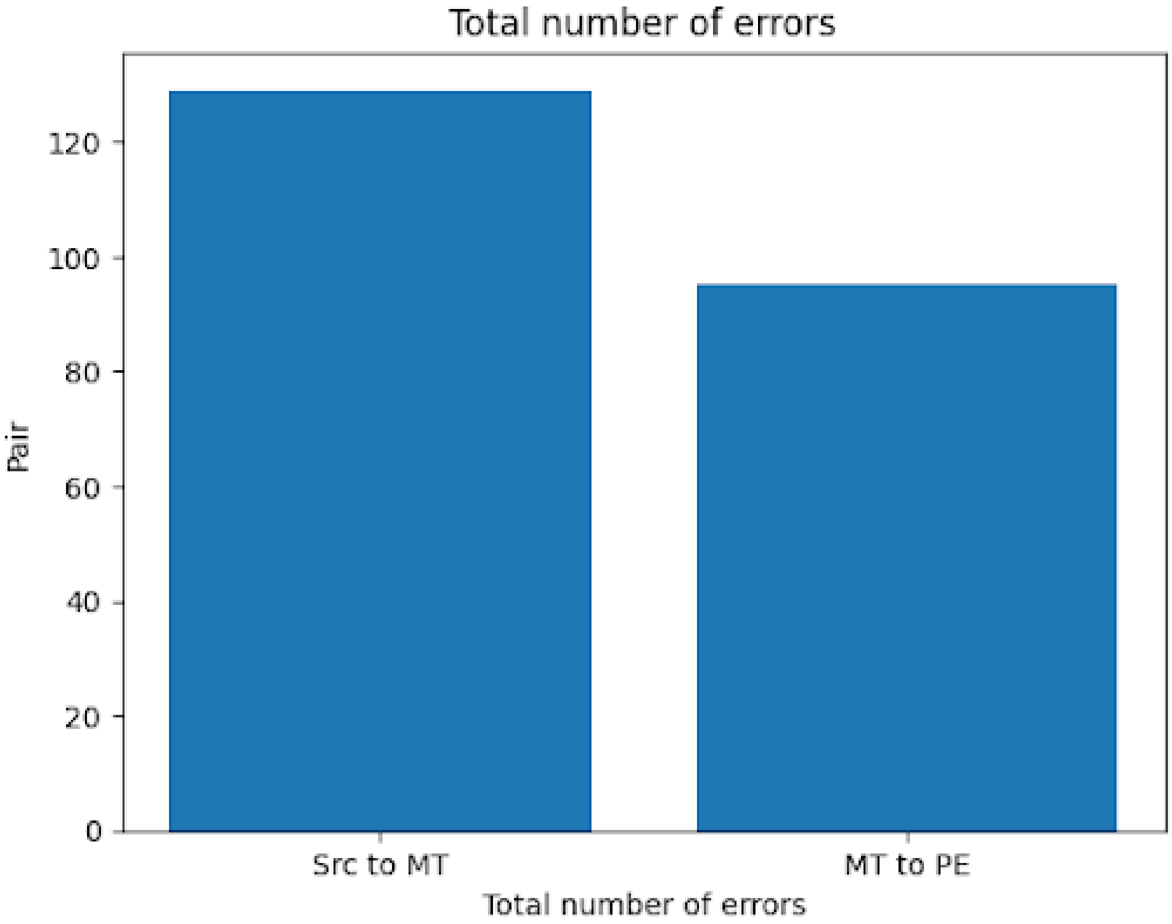

Figure 5.4: Total Number of errors

However, regardless of the proximity of these two counts, when calculating the total severity across all errors, there is a difference of almost 2x between MT and APE. That is, if we add the severity of all 124 errors in MT vs the same sum across all 96 errors in APE, the difference in those scores is now 163 to 92 respectively. So while there is only a ∼22% difference in the number of errors between MT and APE, the difference in the sum of severities is ∼43%.

Here, while it would be correct to not associate severity with errors, the equal distribution of errors and yet decrease in severity does lend credibility to the fact that MT has more severe errors, which very much need post-editing while PE data has errors of less magnitude, most which can be passed on without changing.

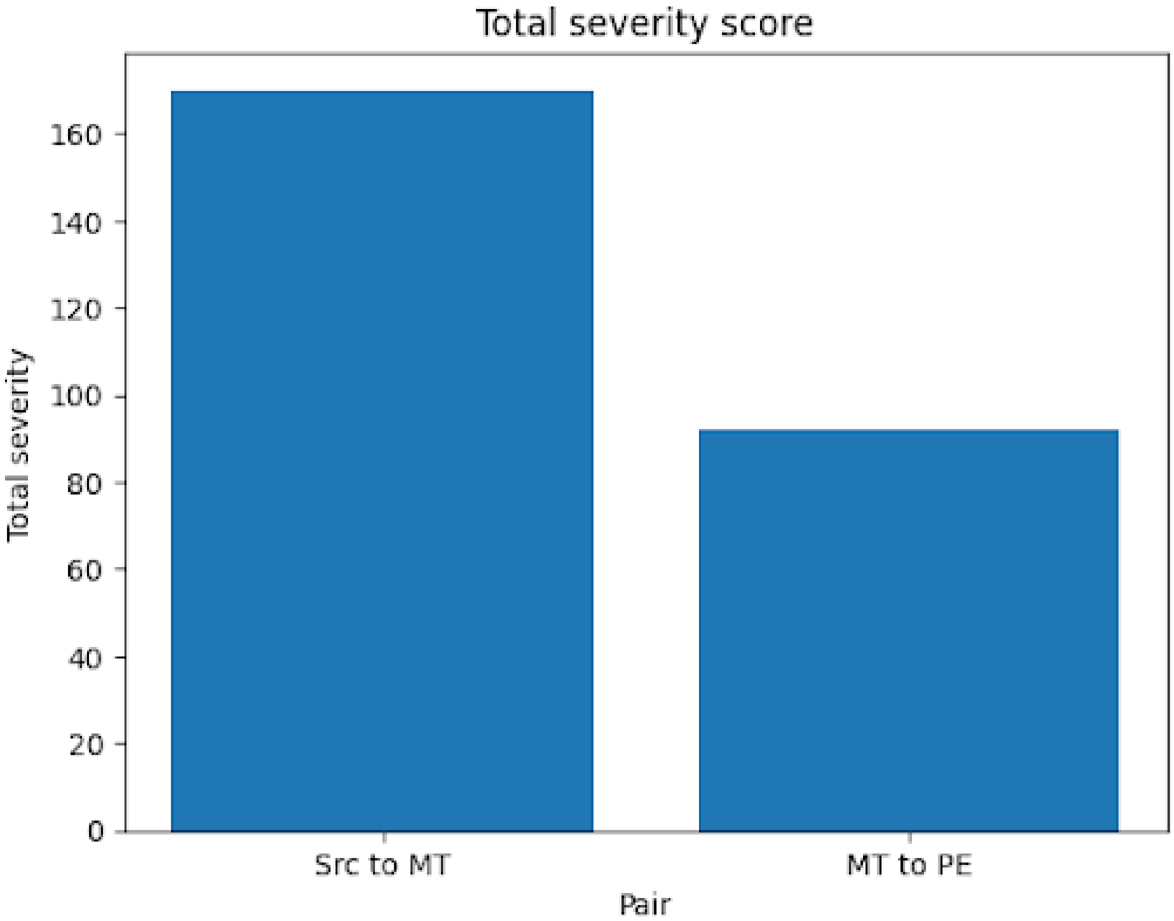

Figure 5.5: Total severity sum across all errors

This could be reasoned to the fact that post-APE, despite the persistence of errors, the average severity of each error is significantly lower for APE, showing higher quality sentence construction than that of an average sentence from MT.

This conclusion is further confirmed when we compare the frequency of severity levels across the two comparisons:

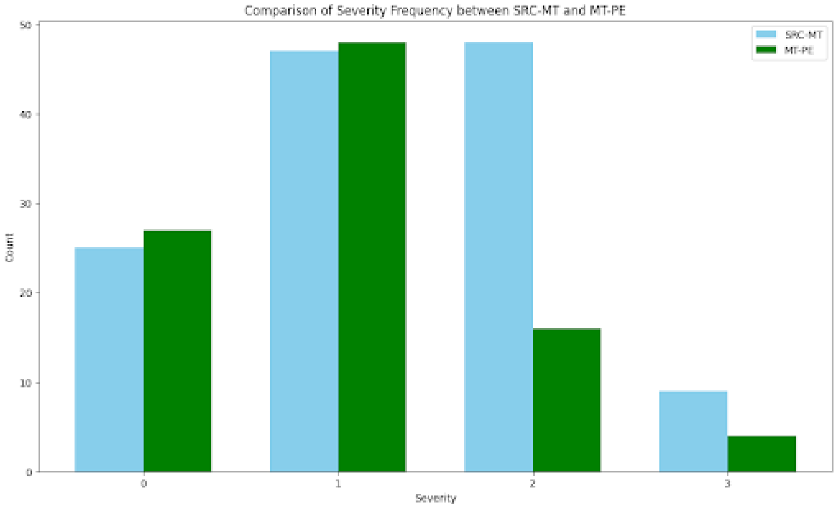

Figure 5.6: Counts of each severity level

As the diagram shows, while MT and APE have almost equal numbers of errors with a severity of 0 or 1, APE has nearly 2.5x the number of errors with severity 2 and 2x the number of errors with severity 3. This clearly indicates that the quality of text has indeed increased post-APE as errors with high severity in MT are removed, and new errors introduced or sustained from MT to APE are mostly of low severity. The ratio of average severities in MT and APE rounds up to 1.3:0.96 respectively.

Let's now move on to discussing the types of errors identified and what they represent. The most common error type in MT was ID 5, attributed to simple mistranslation. This caters only to those errors where the translation is completely wrong and in no way resembles the required/expected reference translation. On the other hand, the most common ID in PE is '-1'. This was a label assigned to MT-PE or SRC-MT pairs where there were no identifiable errors. If a pair had -1, it did not possess any other error, basically implying they were majorly error-free and not requiring post-editing. This shows that the largest portion of PE sentences did not require Post Editing of any extent. To further study this comparison, let's plot a graph comparing the frequencies of types of errors in MT as opposed to those in APE.

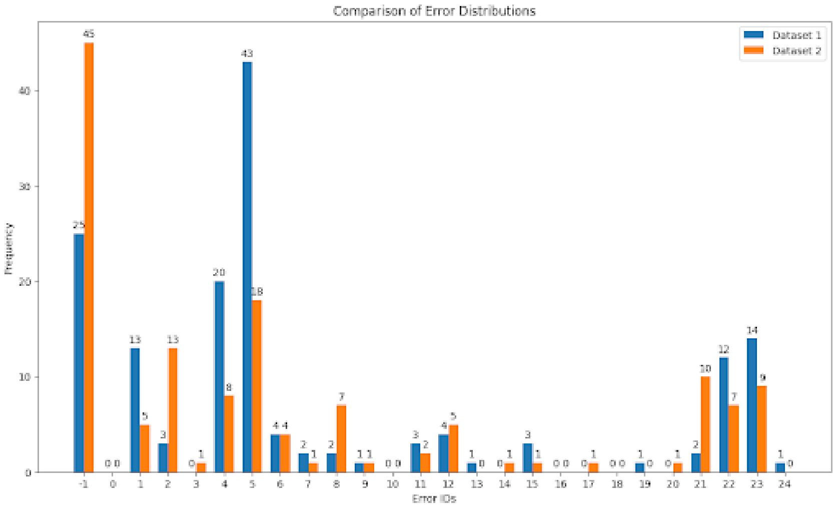

Figure 5.7: Counts of each error type in MQM

In the above figure 5.7, Dataset 1 represents SRC-MT and Dataset 2 represents MT-APE.

Here are the most tagged MQM IDs/errors found across all MT sentences, in descending order of frequency: 5, -1, 4, 23, 1.

Here are the most tagged MQM IDs/errors found across all APE sentences, in descending order of frequency: -1, 5, 2, 21, 23.

Here is some basic analysis done on the same:

5.5.3 Overall Error Counts

- **MT (Dataset 1)**: High error counts are observed in IDs -1, 5, and 7.
- **APE (Dataset 2)**: There is a notable reduction in errors for IDs -1, 5, and 11 but an increase in IDs 2, 21, and 23.

5.5.4 Key Error Types

Terminology Errors (IDs 1-4)

- Significant improvements are seen in ID 1 (Inconsistent with terminology resource), where APE eliminated the errors present in MT.
- IDs 2 and 4 show a moderate reduction in errors post-APE.

Accuracy Errors (IDs 5-11)

- ID 5 (Mistranslation) is significantly reduced by APE.
- IDs 6 and 7 (Overtranslation and Undertranslation) show slight reductions.
- ID 11 (Untranslated) errors are completely eliminated by APE.

Linguistics Conventions (IDs 12-17)

- ID 12 (Grammar) shows a notable reduction post-APE.
- IDs 13, 14, and 15 remain low in both datasets.

Style Errors (IDs 18-24)

- ID 21 (Language Register) and ID 23 (Unidiomatic Style) show increases in APE, indicating new style-related issues introduced post-editing.

5.5.5 Effectiveness of APE

- APE significantly reduces mistranslation errors and completely resolves untranslated content issues.
- Grammatical improvements are notable, showing APE's ability to handle syntactic corrections effectively.

5.5.6 Persistent Issues

- Certain errors like under translation, terminology inconsistencies, and omission remain persistent, indicating areas where APE needs further refinement.
- The increase in errors related to language register and unidiomatic style suggests that while APE improves certain aspects, it may sometimes introduce new stylistic issues, making sentences sound unnatural or inconsistent.

5.5.7 Potential Improvements

- Focus on refining APE models to handle under translation and omission errors more effectively.
- Enhance terminology consistency handling within the APE system to reduce inconsistent use and wrong term errors.
- Incorporate advanced stylistic and register-consistency checks within APE to minimize the introduction of new stylistic errors.

5.6 Experiments for WMT Marathi Shared Task

Having experimented with neural architectures and data augmentation, I decided to participate in the WMT '23 Marathi Automatic Post-Editing Shared Task, which was focused on advancing the quality of machine-generated translations in Marathi through automated post-editing techniques. Building upon their shared task of 2022, this initiative aimed to address the nuanced challenges inherent in Marathi translation, including syntax, semantics, and cultural nuances. Participants were provided with machine-translated text and tasked with developing algorithms and systems to automatically refine and improve these translations. The previous study and the rigorous evaluation of its submissions and results were benchmarked against established metrics.

5.6.1 Dataset

The dataset is composed of 18,000 triplets of source (English), target (Hindi), and Post-Edited (Hindi) data. The data is created by taking a parallel corpus, where the source data is translated using an MT system, and the references are considered post-edits. Its a combination of 18 independent datasets over 8 different domains - Entertainment, Business, General, Medical, Legal, News, Sports, and Tech.

5.6.2 Experiments

Having performed this study post the Hindi post-editing efforts that are mentioned earlier in this chapter, the same neural data augmentation methodologies were reproduced for this study.

WMT ’23 Marathi APE Shared Task			
Data	**BLEU score**	**CHRF**	**TER**
Baseline Translation [116]	70.66	79.78	26.6

Table 5.7: Baseline score for the WMT ’23 Marathi APE Shared Task

Experiments with Synthetically Generated Data (IIT-B)					
Training Data	**Architecture**	**Train Dataset Size (Number of Triplets)**	**BLEU score**	**TER**	**CHRF**
HiEn-Train	DET2T-PE (Cross-Attention)	18,000	60.85	0.26	79.78
HiEn-Train + FWG	DET2T-PE (Cross-Attention)	36,000	67.86	0.12	84.45
HiEn-Train + BWG	DET2T-PE (Cross-Attention)	36,000	64.48	0.21	81.21
HiEn-Train + BWT	DET2T-PE (Cross-Attention)	36,000	37.71	0.56	63.00

Table 5.8: Scores when training data is augmented using various neural methodologies, on WMT Marathi data

The results found are interesting but not out of place when compared to other submissions made to this shared task. As observed in the study with Hindi data, augmenting and adding

FWG to the original data brings us the highest performance. The highest scores amongst all the best-performing submissions (excluding late submission) range maxes out at 69.03 - which implies (and is stated in the paper as being true) that none of the submissions outdid the baseline. This difference in scores could be attributed to a slew of reasons. For once, the study allowed participants to use pre-trained language models as well as use any external dataset available. When implementing it ourselves, we stuck to using only the provided text and trained the language model from scratch.

As mentioned, none of the on-time accepted solutions were able to improve on this score either. The highest BLEU score of all submissions was 69.03 while the lowest was 31.63 [116]. Unfortunately, even though we scored 67 BLEU using my model, due to a logistics issue, we were unable to submit the final data on time.

Chapter 6

The Role of Large Language Models in Automatic Post-Editing

Chapter 6 delves into the significant impact of large language models (LLMs) on the domain of automatic post-editing (APE). It begins with an introduction to LLMs, highlighting their advancements in natural language processing (NLP) and artificial intelligence (AI). The chapter explores how these models, through their deep learning algorithms and extensive training on vast datasets, have revolutionized text generation, translation, summarization, and question-answering tasks. It further examines the integration of LLMs into APE, emphasizing the transformative shift from rule-based and statistical approaches to sophisticated deep learning techniques. The experiments conducted with LLMs in APE are detailed, showcasing their enhanced capabilities in refining machine-translated texts. Finally, the chapter concludes with insights and future directions, setting the stage for the continued evolution of APE powered by LLMs.

6.1 Introduction to Large Language Models (LLMs)

Large Language Models (LLMs) have become central to Automatic Post-Editing (APE) in Natural Language Processing (NLP), marking a significant evolution from rule-based and statistical approaches to leveraging deep learning models for refining machine-translated texts, thereby improving their quality, fluency, and accuracy.

The journey was arguably kick-started with ”Attention Is All You Need” [29], which introduced the Transformer architecture, a novel approach that eschews the recurrent layers used in previous models for an attention mechanism, enabling significantly improved efficiency and scalability in processing language data. Subsequent influential works include the seminal paper that introduced BERT [117], a model that leverages the Transformer to understand language context in a deeper and more nuanced manner than was previously possible. BERT's methodology of pre-training on a large corpus and fine-tuning for specific tasks revolutionized how models are developed and deployed in NLP applications.

Furthermore, the GPT series [118] showcased the potential of generative pre-trained models in producing coherent and diverse text, pushing the boundaries of what's possible in text generation, translation, and more. For example, GPT-3 [119], with its 175 billion parameters, can compose essays, translate languages, and even generate code, given appropriate prompts. BERT [117], specializing in understanding context, significantly improves search results by understanding the intent behind queries. These models serve various purposes, from powering chatbots and virtual assistants to enhancing search engines and automating content creation.

Model	**Parameters (In Millions)**	**Training Data**
GPT-3	175,000	Diverse web text
T5	11,000	C4
ELECTRA	335	Wikipedia, BooksCorpus
RoBERTa	355	Larger subset of web text
GPT-2	1500	Diverse web text

Table 6.1: Comparison of some common Large Language Models and Pre-trained Language Models

6.2 Zero-shot Prompting

Automatic post-editing focuses on refining the output of machine translation systems or other generative models to correct errors or improve fluency, often employing models like BERT or

GPT-3 to automatically revise text towards a target style or higher quality [117, 119] . APE benefits from zero-shot prompting by using prompts to guide the correction process, effectively merging the understanding encapsulated within LLMs with specific correction goals, without the need for task-specific model retraining.

The synergy between zero-shot prompting and APE in the context of LLMs illustrates a powerful method for enhancing text quality and accuracy in a wide range of applications, from content creation to translation. This integrated approach highlights the evolving landscape of NLP, where adaptable, pre-trained models like GPT-3 and BERT can be dynamically applied to both understand and improve human language in real-time, marking a significant step forward in achieving more natural and efficient human-computer interaction.

6.3 LLMs in Automatic Post-Editing

Large Language Models (LLMs) have increasingly become central to the field of Automatic Post-Editing (APE), a niche yet vital area within Natural Language Processing (NLP) that focuses on refining machine-translated texts to improve their quality, fluency, and accuracy. The integration of LLMs into APE processes marks a significant evolution from earlier rule-based and statistical approaches to leveraging deep learning models for understanding and improving translations. This transition has been facilitated by seminal works and the development of advanced LLMs that have progressively enhanced the capabilities of APE systems.

LLMs boost the field of APE by enabling context-aware editing of texts. BERT [117] and its successors, like RoBERTa [120], showcased the power of bidirectional context understanding in text, laying the groundwork for more sophisticated APE systems. These models, pre-trained on vast amounts of text data, could be fine-tuned for APE tasks, significantly improving the post-editing process by understanding the nuances of language better than ever before. The introduction of GPT-2 [118] and GPT-3 [119] by OpenAI further revolutionized the field by demonstrating how generative models could be applied to a wide range of language tasks, including APE. Their ability to generate human-like text and adapt to various language styles and formats opened new avenues for automatic post-editing, making it possible to correct and enhance translated texts more efficiently.

The application of LLMs in APE has also been explored in academic studies focusing on specific aspects of post-editing, such as grammatical error correction [121] and style adaptation [122]. These studies have highlighted the versatility of LLMs in addressing the multifaceted challenges of APE, from correcting simple typographical errors to adjusting the tone and style of translated texts to match target audience preferences. Exploring APE as an overarching topic, this work by Vikas et al. [123].

Moreover, the emergence of domain-specific LLMs has further refined the APE process. For instance, models trained on specialized corpora, such as legal or medical texts, have demonstrated improved performance in post-editing tasks within these domains [124]. This specialization underscores the adaptability of LLMs to the nuanced requirements of different fields, enhancing the quality of machine-translated texts across various industries.

The evolution of LLMs in APE is not just limited to improving individual aspects of the post-editing process but also encompasses the integration of these models into end-to-end translation workflows [125]. This holistic approach leverages LLMs for both initial translation and subsequent editing, streamlining the translation process and ensuring a higher quality of the final text.

Additionally, the versatility of LLMs and their understanding of knowledge can now enable a series of supplementary tasks associated with Automatic Post Editing. For example, executing the task of checking if a text warrants the time and computational power/effort to post-edit is an aspect that can be checked by LLMs [126]. Most popular LLMs are also language agnostic. Given that they are trained on data across the internet rather than data classified by language, most modern LLMs like ChatGPT-4 are multi-lingual and hence can use both the source as well as target data in an APE task to assess the quality of the APE system [127].

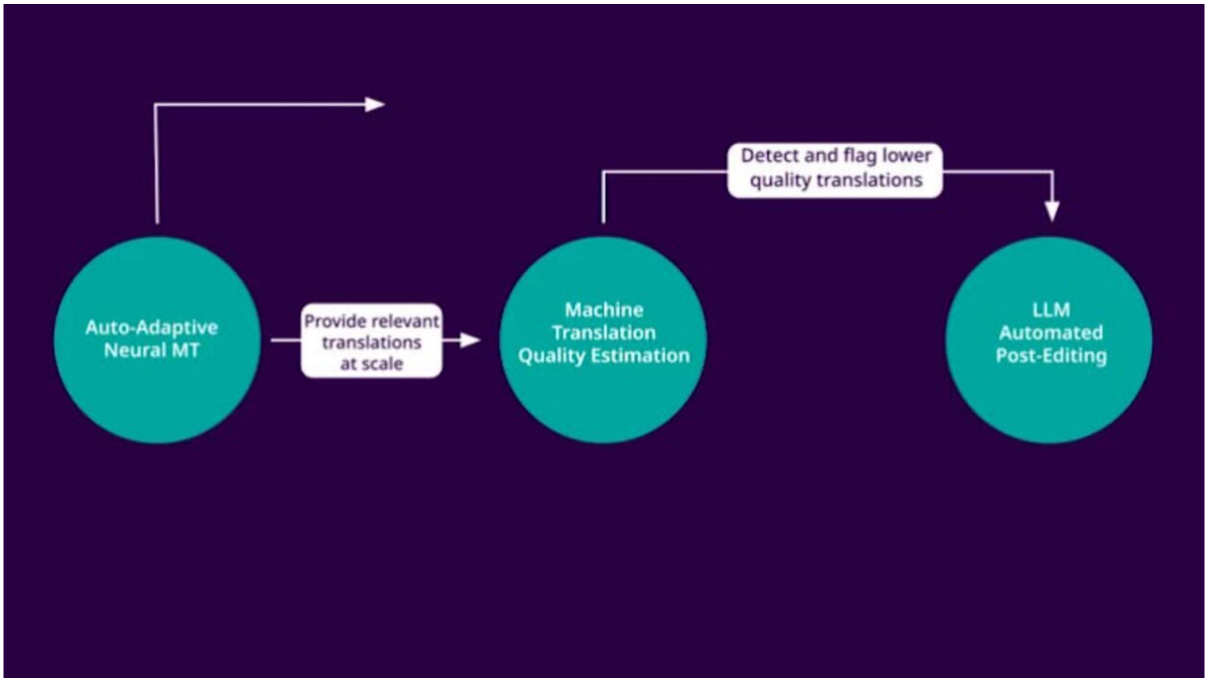
Auto-Adaptive Neural MT
Provide relevant translations at scale
Machine Translation Quality Estimation
Detect and flag lower quality translations
LLM Automated Post-Editing

Figure 6.1: Use of LLMs to estimate quality of translation, checking if APE is warranted [1]

6.4 Experiments

LLMs - in our context, specifically ChatGPT-3.5, have been studied thoroughly in the context of Machine Translation [128]. While there has been work done in leveraging LLMs for Automatic Post Editing [123], no work has explored using FLORES200 and NTREX datasets to study Automatic POst Editing of Hindi sentences translated from English. By conducting experiments with FLORES and NTREX test datasets, this study aims to uncover insights into how GPT-3.5.0 can be utilized to improve the quality of MT outputs significantly.

6.4.1 Setup

For our choice of LLM, we picked GPT-3.5, the latest language model published by OpenAI. For the purpose of this experiment, we use the ChatGPT-3.5 API. In specific, we use the Text Completions endpoint, which paired with our API key, is fed in as a request in the call. As the first step, we use the src_en and nmt_hi datasets, alongside the prompt template in order to

[1] https://www.rws.com/blog/the-evolution-of-translation/.

get the model to perform post editing. I iterate through the entire dataset and for each pair of src_en-nmt_hi data points, I give the following prompt as an input to the GPT-3.5 Model:

> ”Here is my English source sentence: ” + <insert src_en> + ”, here is my Hindi machine translated sentence of this English sentence: ” + <insert nmt_hi> + ”. With these 2 inputs, give me a post-edited hindi sentence (where you are post-editing the machine translated Hindi text I provided)”

6.4.2 Results

Scores for APE output (FLORES200)				
APE Model	**BLEU score**	**CHRF**	**TER**	**COMET**
Baseline Translation (LTRC-NMT)	33.45	0.58	51.81	0.71
GPT-3.5	25.45	0.54	0.42	0.77
DET2T-PE (Cross-Attention)	23.91	0.53	0.49	0.68

Table 6.2: Comparing machine translation scores from ChatGPT against our State-of-the-Art model

The difference in quality of metrics over the FLORES200 is conclusive but by a very narrow margin. The initial thesis was the GPT-3.5 would be able to perform substantially better than the locally trained DET2T-PE (Cross Attention) model. The results are in line with these expected observations, however, the GPT-3.5 model beats the locally trained model by a very small amount - with ChatGPT presenting a BLEU score that is better by 6%, a 1% improvement in the CHRF score and a 21% difference in the respective TER scores. While the difference is almost insignificant for BLEU and CHRF, there is an unusually large difference between the TER scores of both the studies. A similar difference in scores can be seen between the COMET scores of the two experiments, however, this time the COMET score for GPT-3.5 takes a 13% lead over the COMET score for DET2T-P2 - marking an improved performance by

GPT-3.5 over DET2T-P2 across all metrics. However, with the two studies being almost neck and neck, this result cannot be used conclusively to prove the semantic, syntactic, or pragmatic superiority of the GPT-3.5 model over the DET2T-PE Model.

Here is an example of the different post-editing outputs:

DET2T-PE model (src_en + ape_output + tgt_hi):

- **src_en**: Hsieh implied during the election that Ma might flee the country during a time of crisis.
- **nmt_hi**: हसीह ने चुनाव के दौरान आरोप लगाया कि मा संकट के समय देश से भाग सकता है।
- **tgt_hi**: चुनाव के दौरान साइह ने जताया कि संकट के दौर में मा देश छोड़कर भाग सकते हैं.
- **ape_hi**: हसीह ने चुनाव के दौरान आरोप लगाया कि मा संकट के समय देश से भाग सकता है।

GPT-3.5 model (src_en + ape_output + tgt_hi):

- **src_en**: Hsieh implied during the election that Ma might flee the country during a time of crisis.
- **nmt_hi**: हसीह ने चुनाव के दौरान आरोप लगाया कि मा संकट के समय देश से भाग सकता है।
- **tgt_hi**: चुनाव के दौरान साइह ने जताया कि संकट के दौर में मा देश छोड़कर भाग सकते हैं.
- **ape_hi**: चुनाव के दौरान हसीह ने संकेत दिया कि संकट के समय मा देश छोड़ सकते हैं।

The difference in metric scores for NTREX dataset narrates a different story. In every metric - apart from COMET - DET2T-PE is shown to perform better. However, the margin of improvement seen in the NTREX study very closely mirrors the margin of improvement of scores seen in the corresponding study involving the FLORES200 dataset. However, unlike the large difference seen in the TER scores of the FLORES200 screen, all metrics in the NTREX study seem to have a uniform difference in scores of <=5%.

Here is an example of the different post-editing outputs:

DET2T-PE model (src_en + ape_output + tgt_hi):

Scores for APE output (NTREX200)				
APE Model	**BLEU score**	**CHRF**	**TER**	**COMET**
Baseline Translation (LTRC-NMT)	25.69	0.51	60.00	0.61
GPT-3.5	18.89	46.27	67.73	0.73
DET2T-PE (Cross-Attention)	20.87	0.477	65.03	0.67

Table 6.3: Comparing APE scores of ChatGPT against our State-of-the-Art model

- **src_en**: ”Most of these headliners appeared at Carolines, and went on to greater success, to the point where theyŕe too big to play a club,” Hirsch said.
- **nmt_hi**: इनमें से अधिकांश हेडलाइनर कैरोलिन में दिखाई दिए, और अधिक सफलता के लिए चले गए, इस बिंदु पर कि वे एक क्लब खेलने के लिए बहुत बड़े हैं, हिर्श ने कहा।
- **tgt_hi**: "लोगों को देश भविष्य के लिए जिम्मेदार लोगों को सरकार और राष्ट्रपति हैं, और लगता है कि वह समस्याओं के लिए समाधान को खोजने के लिए एक होना चाहिए, "वे कहते हैं.
- **ape_hi**: हिर्श कहती हैं, इनमें से अधिकांश हेडलाइनर्स कैरोलाइन्स में दिखीं, तथा बहुत बड़ी सफलता बन गईं, ऐसे स्तर तक जहां पर वे क्लब में प्ले किए जाने के लिए बहुत बड़ी बन गईं ।

GPT-3.5 model (src_en + ape_output + tgt_hi):

- **src_en**: ”Most of these headliners appeared at Carolines, and went on to greater success, to the point where theyŕe too big to play a club,” Hirsch said.
- **nmt_hi**: इनमें से अधिकांश हेडलाइनर कैरोलिन में दिखाई दिए, और अधिक सफलता के लिए चले गए, इस बिंदु पर कि वे एक क्लब खेलने के लिए बहुत बड़े हैं, हिर्श ने कहा।
- **tgt_hi**: "लोगों को देश भविष्य के लिए जिम्मेदार लोगों को सरकार और राष्ट्रपति हैं, और लगता है कि वह समस्याओं के लिए समाधान को खोजने के लिए एक होना चाहिए, "वे कहते हैं.
- **ape_hi**: हिर्श ने कहा, "इन हेडलाइनर में से अधिकांश कैरोलाइन्स पर दिखाई देने वाले थे, और उन्होंने बड़ी सफलता हासिल की, ऐसा कि वे किसी क्लब में खेलने के लिए बहुत बड़े हो गए हैं।

Scores for APE output (IIT-B)				
APE Model	**BLEU score**	**CHRF**	**TER**	**COMET**
GPT-3.5	22.78	0.51	65.61	0.80
DET2T-PE (Cross-Attention)	27.26	0.53	54.60	0.73

Table 6.4: Comparing English to Hindi NMT using LTRCNMT and GPT-3.5

However, a consistent trend across all 3 experiments involves the COMET score of GPT-3.5 being notably higher than the COMET score of the corresponding DET2T-PE model. This difference can be anywhere between 8% to 20%. This could be due to the fact that GPT-3.5 is primarily trained on English data, with a small focus on Hindi. Due to this, the study involving semantic understanding of English to its Hindi counterpart could sway the translation to be more semantically led. BLEU, TER, and CHRF focus more on surface-level linguistic similarities, such as n-gram overlap or edit distance, where localized training can tightly align with specific translation tasks. In contrast, COMET, designed to assess semantic adequacy and fluency using advanced language models, might favor GPT-3.5's extensive knowledge and nuanced understanding of English. This suggests GPT-3.5 excels in capturing deeper, contextual language relationships that COMET evaluates, highlighting the importance of considering multiple metrics for a holistic assessment of translation quality.

6.5 Conclusion

In summary, the use of LLMs in the field of Automatic Post-Editing has undergone significant evolution, moving from foundational models that introduced new architectural paradigms to advanced generative models capable of producing high-quality, contextually accurate text. The ongoing research and development in this area continue to push the boundaries of what is possible, promising even more sophisticated and efficient APE systems in the future. This evolution, documented through a series of influential papers and studies, highlights the transformative impact of LLMs on the APE field, setting a trajectory for continued innovation and improvement.

The integration of LLMs in APE workflows is increasingly common, with ongoing research exploring optimal training and application methods to enhance translation quality further. The dynamic nature of LLMs, combined with their scalability, makes them well-suited to adapt to the evolving demands of language translation and editing tasks. This result yet again enforces another conclusion made previously in this study that shows that a model trained specifically for the purpose of Automatic Post Editing - preferbally with homogenous data - is more likely to perform better than a generic language model.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis presents a comprehensive study on the application of advanced techniques in automatic post-editing (APE) for English-Hindi translations. By leveraging neural models and data augmentation strategies, we sought to address the inherent challenges in improving machine translation (MT) outputs. Our approach integrates various methodologies, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), and Transformer-based models, to refine the quality of Hindi translations.

The experimental results demonstrated that neural models, particularly those employing Transformer architectures, significantly outperform traditional statistical methods in APE tasks. The Transformer models excel in handling long-range dependencies and variable-length sequences, which are crucial for accurately capturing the complexities of the Hindi language. The Dual-Encoder Transformer model, in particular, showed remarkable improvements in BLEU, CHRF, TER, and COMET scores, highlighting its effectiveness in post-editing tasks.

This study also underscored the importance of data augmentation in enhancing model performance. Techniques such as forward generation, backward generation, and backward translation were instrumental in creating a robust training dataset, addressing the scarcity of triplet data required for APE. These methods not only expanded the dataset but also introduced variability, enabling the models to generalize better and improve translation accuracy.

Moreover, we explored the integration of Large Language Models (LLMs) and Pre-trained Language Models (PLMs) in APE. These models, through zero-shot and few-shot learning, demonstrated potential in refining translation outputs without extensive retraining, thus offering a scalable solution for real-world applications.

7.2 Future Work

While this research has made significant strides in advancing APE for English-Hindi translations, several avenues remain unexplored and warrant further investigation:

1. **Enhanced Data Augmentation Techniques:** Future research could explore more sophisticated data augmentation techniques, including adversarial training and synthetic data generation using Generative Adversarial Networks (GANs) [129]. These methods could introduce greater diversity in training datasets, further improving model robustness and translation quality.

2. **Multimodal APE:** Incorporating multimodal data, such as visual and auditory inputs, could enhance the contextual understanding of translations [130]. This approach, known as multimodal neural machine translation (MNMT) [131], has shown promise in handling low-resource languages and could be particularly beneficial for complex language pairs like English-Hindi.

3. **Real-time APE Systems:** Developing real-time APE systems that can integrate seamlessly with existing MT frameworks is another critical area for future work. Such systems would require optimizing model architectures for low-latency inference, ensuring that the post-editing process does not introduce significant delays in translation workflows.

4. **Domain-Specific APE Models:** Tailoring APE models to specific domains, such as legal, medical, or technical translations, could improve accuracy and relevance. Domain-specific models can leverage specialized terminologies and contextual knowledge, addressing the unique challenges posed by different fields.

5. **Human-in-the-Loop APE:** Integrating human feedback into the APE process could enhance model training and evaluation. A human-in-the-loop approach allows for continuous improvement of the models by incorporating real-time corrections and suggestions from human post-editors, thereby improving the quality and reliability of translations.

6. **Cross-lingual Transfer Learning:** Applying transfer learning techniques across different language pairs could help in developing more versatile APE models. By leveraging knowledge from high-resource language pairs, such as English-Spanish, to improve low-resource pairs like English-Hindi, models can benefit from shared linguistic features and improve overall performance.

7. **Ethical and Bias Considerations:** Future research should also focus on addressing ethical considerations and mitigating biases in APE models. Ensuring that translations are fair, unbiased, and culturally sensitive is crucial for building trustworthy MT systems.

In conclusion, this thesis has laid the groundwork for advanced APE techniques in English-Hindi translations, showcasing the potential of neural models and data augmentation. However, the field remains ripe with opportunities for further innovation and improvement. By addressing the identified future work areas, researchers can continue to enhance the quality, efficiency, and applicability of APE systems, contributing to more accurate and fluent machine translations for diverse languages and contexts.

7.2.1 Examiner Comments

In the current thesis, the experiments of APE are trivial.

This is so because the main objective of this study is to study data augmentation models and see their response to state-of-the-art models.

Also, the data augmentation methods could be more strategic.

The data augmentation methods were indeed strategic in nature. The reason we picked Forward generation, Backward Generation, and Back-translation is because these 3 augmentation methods only involve changing ready existing data like the input or output to the APE model.

Additional examples have also been added.

Bibliography

[1] Iñaki Alegria, Unai Cabezon, Unai Fernández de Betoño, Gorka Labaka, Aingeru Mayor, Kepa Sarasola, and Arkaitz Zubiaga. *Reciprocal Enrichment Between Basque Wikipedia and Machine Translation*, pages 101–118. 02 2013.

[2] WonKee Lee, Junsuk Park, Byung-Hyun Go, and Jong-Hyeok Lee. Transformer-based automatic post-editing with a context-aware encoding approach for multi-source inputs. *ArXiv*, abs/1908.05679, 2019.

[3] Rajen Chatterjee, Marion Weller, Matteo Negri, and Marco Turchi. Exploring the planet of the APEs: a comparative study of state-of-the-art methods for MT automatic post-editing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 156–161, Beijing, China, July 2015. Association for Computational Linguistics.

[4] Joss Moorkens Joachim Wagner Murhaf Hossari Eric Paquin Dag Schmidtke Declan Groves Andy Way Félix do Carmo, Dimitar Shterionov. A review of the state-of-the-art in automatic post-editing. *Machine Translation volume*, pages 101 – 143, 2021.

[5] Sharon O'Brien. Introduction to post-editing: Who, what, how and where to next? In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Tutorials*, Denver, Colorado, USA, October 31-November 4 2010. Association for Machine Translation in the Americas.

[6] Ren Wu Qiming Chen. Cnn is all you need. *rXiv:1712.09662v1*, 2017.

[7] Kristen Parton, Nizar Habash, Kathleen McKeown, Gonzalo Iglesias, and Adrià de Gispert. Can automatic post-editing make MT more meaningful. In *Proceedings of the 16th Annual conference of the European Association for Machine Translation*, pages 111–118, Trento, Italy, May 28–30 2012. European Association for Machine Translation.

[8] Sitender, Seema Bawa, Munish Kumar, and Sangeeta. A comprehensive survey on machine translation for english, hindi and sanskrit languages. *Journal of Ambient Intelligence and Humanized Computing*, Sep 2021.

[9] Ondřej Bojar, Vojtěch Diatka, Pavel Rychlý, Pavel Straňák, Vít Suchomel, Aleš Tamchyna, and Daniel Zeman. HindEnCorp - Hindi-English and Hindi-only corpus for machine translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3550–3555, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[10] Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. The iit bombay english-hindi parallel corpus, 2017.

[11] B. Premjith, M. Anand Kumar, and K.P. Soman. Neural machine translation system for english to indian language translation using mtil parallel corpus. *Journal of Intelligent Systems*, 28(3):387–398, 2019.

[12] Colin P. Masica. *The Indo-Aryan Languages.* Cambridge University Press, Cambridge, UK, 1991.

[13] Pattisapu Nikhil Priyatam, Srikanth Reddy Vaddepally, and Vasudeva Varma. Domain specific search in indian languages. In *Proceedings of the First Workshop on Information and Knowledge Management for Developing Region*, IKM4DR '12, page 23–30, New York, NY, USA, 2012. Association for Computing Machinery.

[14] Sahinur Rahman Laskar, Abdullah Faiz Ur Rahman Khilji, Partha Pakray, and Sivaji Bandyopadhyay. Multimodal neural machine translation for English to Hindi. In Toshiaki Nakazawa, Hideki Nakayama, Chenchen Ding, Raj Dabre, Anoop Kunchukuttan, Win Pa Pa, Ondřej Bojar, Shantipriya Parida, Isao Goto, Hidaya Mino, Hiroshi Manabe, Katsuhito Sudoh, Sadao Kurohashi, and Pushpak Bhattacharyya, editors, *Proceedings of*

the 7th Workshop on Asian Translation, pages 109–113, Suzhou, China, December 2020. Association for Computational Linguistics.

[15] Sahinur Rahman Laskar, Abdullah Faiz Ur Rahman Khilji, Darsh Kaushik, Partha Pakray, and Sivaji Bandyopadhyay. Improved English to Hindi multimodal neural machine translation. In Toshiaki Nakazawa, Hideki Nakayama, Isao Goto, Hideya Mino, Chenchen Ding, Raj Dabre, Anoop Kunchukuttan, Shohei Higashiyama, Hiroshi Manabe, Win Pa Pa, Shantipriya Parida, Ondřej Bojar, Chenhui Chu, Akiko Eriguchi, Kaori Abe, Yusuke Oda, Katsuhito Sudoh, Sadao Kurohashi, and Pushpak Bhattacharyya, editors, *Proceedings of the 8th Workshop on Asian Translation (WAT2021)*, pages 155–160, Online, August 2021. Association for Computational Linguistics.

[16] Xixuan Huang, Nankai Lin, Kexin Li, Lianxi Wang, and Suifu Gan. Hinplms: Pretrained language models for hindi. In *2021 International Conference on Asian Language Processing (IALP)*, pages 241–246, 2021.

[17] A simple and effective approach to automatic post-editing of machine translation. *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing.*

[18] Yehoshua Bar-Hillel. The present status of automatic translation of languages. *Adv. Comput.*, 1:91–163, 1960.

[19] Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, and Josef van Genabith. A neural network based approach to automatic post-editing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 281–286, Berlin, Germany, August 2016. Association for Computational Linguistics.

[20] Spence Green, Jeffrey Heer, and Christopher D. Manning. The efficacy of human post-editing for language translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 439–448, New York, NY, USA, 2013. Association for Computing Machinery.

[21] Siddharth Dalmia Yuya Fujita Shinji Watanabe Motoi Omachi, Brian Yan. Align, write, re-order: Explainable end-to-end speech translation via operation sequence generation. *arXiv preprint arXiv:2107.05899*, 2021.

[22] Liling Tan Ewa Szymanska Shamil Chollampatt, Raymond Hendy Susanto. Can automatic post-editing improve nmt? *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 2736–2746, 2020.

[23] Aleksandar Savkov Ehud Reiter Francesco Moramarco, Alex Papadopoulos Korfiatis. A preliminary study on evaluating consultation notes with post-editing. *MProceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, page 62–68, 2021.

[24] Siyu Zou. Analysis of machine translation and post-translation editing ability using semantic information entropy technology. *Journal of Environmental and Public Health*, 2022:5932044, Aug 2022.

[25] Hyeonseok Moon; Chanjun Park; Jaehyung Seo; Sugyeong Eo; Heuiseok Lim. An automatic post editing with efficient and simple data generation method. *MProceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 21032 – 21040, 2022.

[26] WonKee Lee, Seong-Hwan Heo, Baikjin Jung, and Jong-Hyeok Lee. Towards semi-supervised learning of automatic post-editing: Data-synthesis by infilling mask with erroneous tokens, 2022.

[27] Kevin Knight and Ishwar Chander. Automated postediting of documents. *ArXiv*, abs/cmp-lg/9407028, 1994.

[28] Karen Kukich. Techniques for automatically correcting words in text. *ACM Comput. Surv.*, 24(4):377–439, dec 1992.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[30] Joan L.G. Baart and Vincent J. van Heuven. From text to speech; the mitalk system: Jonathan allen, m. sharon hunnicutt and dennis klatt (with robert c. armstrong and david pisoni): Cambridge university press, cambridge, 1987. xii+216 pp. £25.00. *Lingua*, 81(2):265–270, 1990.

[31] Joss Moorkens Joachim Wagner Murhaf Hossari Eric Paquin Dag Schmidtke Declan Groves Andy Way Félix do Carmo, Dimitar Shterionov. The memo system. early machine aids to translation: Machine translation. *Translating and the Computer 5. Proceedings of a conference*, 1985.

[32] George Foster, Pierre Isabelle, and Pierre Plamondon. Target-text mediated interactive machine translation. *Machine Translation*, 12(1/2):175–194, 1997.

[33] G. Sampson. The "grammar checker" in microsoft word: A critique. *Literary and Linguistic Computing*, pages 257 – 261, 1993.

[34] Kevin Knight and Ishwar Chander. Automated postediting of documents. In *AAAI Conference on Artificial Intelligence*, 1994.

[35] Automatic post-editing of translations: Rules and statistical machine translation. *Journal of Computational Linguistics.*

[36] Michel Simard, Cyril Goutte, and Pierre Isabelle. Statistical phrase-based post-editing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 508–515, Rochester, New York, April 2007. Association for Computational Linguistics.

[37] WonKee Lee, Junsuk Park, Byung-Hyun Go, and Jong-Hyeok Lee. Transformer-based automatic post-editing with a context-aware encoding approach for multi-source inputs. *ArXiv*, abs/1908.05679, 2019.

[38] Kristen Parton, Nizar Habash, Kathleen McKeown, Gonzalo Iglesias, and Adrià de Gispert. Can automatic post-editing make MT more meaningful. In Mauro Cettolo, Marcello Federico, Lucia Specia, and Andy Way, editors, *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, pages 111–118, Trento, Italy, May 28–30 2012. European Association for Machine Translation.

[39] Kristen Parton, Nizar Habash, Kathleen McKeown, Gonzalo Iglesias, and Adrià de Gispert. Can automatic post-editing make MT more meaningful. In *Proceedings of the 16th Annual conference of the European Association for Machine Translation*, pages 111–118, Trento, Italy, May 28–30 2012. European Association for Machine Translation.

[40] Marcello Federico Nicola Bertoldi. Domain adaptation for statistical machine translation with monolingual resources. *Proceedings of the Fourth Workshop on Statistical Machine Translation*, page 182–189, 2009.

[41] Rudolf Rosa, David Mareček, and Ondřej Dušek. DEPFIX: A system for automatic correction of Czech MT outputs. In Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia, editors, *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 362–368, Montréal, Canada, June 2012. Association for Computational Linguistics.

[42] Rudolf Rosa. Automatic post-editing of phrase-based machine translation outputs. 2013.

[43] Jeffrey Allen and Christopher Hogan. Toward the development of a post editing module for raw machine translation output: A controlled language perspective. In *Third International Controlled Language Applications Workshop (CLAW-00)*, pages 62–71, 2000.

[44] Loïc Dugast, Jean Senellart, and Philipp Koehn. Statistical post-editing on SYSTRAN‘s rule-based translation system. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 220–223, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[45] F. Casacuberta R. Silva E. D´ıaz-de-Liano A.-L. Lagarda, V. Alabau. *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, page 217–220, 2009.

[46] Dimitar Shterionov, Félix do Carmo, Joss Moorkens, Murhaf Hossari, Joachim Wagner, Eric Paquin, Dag Schmidtke, Declan Groves, and Andy Way. A roadmap to neural automatic post-editing: an empirical approach. *Machine Translation*, 34:1–30, 09 2020.

[47] Joss Moorkens Murhaf Hossari Joachim Wagner-Eric Paquin Dag Schmidtke Declan Groves Andy Way Dimitar Shterionov, Félix do Carmo. A roadmap to neural automatic post-editing: an empirical approach. *Machine Translation 34*, page 67–96, 2020.

[48] Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA, August 8-12 2006. Association for Machine Translation in the Americas.

[49] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[50] Joseph Ochlenius, Christoph Vogel, and Hermann Ney. Word error rate (wer) and human evaluation of automatic speech recognition (asr). In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, 2000.

[51] F. Casacuberta R. Silva E. D´ıaz-de-Liano A.-L. Lagarda, V. Alabau. west post-editing of a rule-based machine translation system. *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, page 217–220, 2009.

[52] Shraddha Amit Kalele, Shashi Pal Singh, Prashant Chaudhary, Lenali Singh, Ajai Kumar, and Pulkit Joshi. A hybrid approach towards machine translation system for english–hindi and vice versa. In Yu-Dong Zhang, Tomonobu Senjyu, Chakchai So-In, and Amit Joshi, editors, *Smart Trends in Computing and Communications*, pages 523–532, Singapore, 2023. Springer Nature Singapore.

[53] Omkar Dhariya, Shrikant Malviya, and Uma Shanker Tiwary. A hybrid approach for hindi-english machine translation. 02 2017.

[54] Machine translation post editing (mtpe): A hybrid approach to translation. https://etranslationservices.com/translations/machine-translation-post-editing-mtpe-a-hybrid-approach-to-translation/.

[55] Gabriele Sarti, Arianna Bisazza, Ana Guerberof-Arenas, and Antonio Toral. DivEMT: Neural machine translation post-editing effort across typologically diverse languages. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7795–7816, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[56] Arafat Ahsan, Vandan Mujadia, and Dipti Misra Sharma. Assessing post-editing effort in the English-Hindi direction. In *Proceedings of the 18th International Conference on Natural Language Processing (ICON)*, pages 44–53, National Institute of Technology Silchar, Silchar, India, December 2021. NLP Association of India (NLPAI).

[57] Ltrc translator. https://ltrc.iiit.ac.in/.

[58] Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. The IIT Bombay English-Hindi parallel corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA).

[59] James Cross Onur Çelebi Maha Elbayad-Kenneth Heafield Kevin Heffernan Elahe Kalbassi Janice Lam Daniel Licht Jean Maillard Anna Sun Skyler Wang Guillaume Wenzek Al Youngblood Bapi Akula Loic Barrault Gabriel Mejia Gonzalez Prangthip Hansanti John Hoffman Semarley Jarrett Kaushik Ram Sadagopan Dirk Rowe Shannon Spruit Chau Tran Pierre Andrews Necip Fazil Ayan Shruti Bhosale Sergey Edunov Angela Fan Cynthia Gao Vedanuj Goswami Francisco Guzmán Philipp Koehn Alexandre Mourachko Christophe Ropers Safiyyah Saleem Holger Schwenk Jeff Wang NLLB Team, Marta R. Costa-jussà. No language left+ behind: Scaling human-centered machine translation. 2022.

[60] Christian Federmann, Tom Kocmi, and Ying Xin. NTREX-128 – news test references for MT evaluation of 128 languages. In *Proceedings of the First Workshop on Scaling Up*

Multilingual Evaluation, pages 21–24, Online, nov 2022. Association for Computational Linguistics.

[61] Shashank Singh and Shailendra Singh. Hindia: a deep-learning-based model for spell-checking of hindi language. *Neural Computing and Applications*, 33:3825–3840, 2021.

[62] Samrat Dutta, Shreyansh Jain, Ayush Maheshwari, Souvik Pal, Ganesh Ramakrishnan, and Preethi Jyothi. Error correction in asr using sequence-to-sequence models. *arXiv preprint arXiv:2202.01157*, 2022.

[63] Marcin Junczys-Dowmunt and Roman Grundkiewicz. Ms-uedin submission to the wmt2018 ape shared task: Dual-source transformer for automatic post-editing, 2018.

[64] Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. Findings of the 2017 conference on machine translation (WMT17). In Ondřej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, and Julia Kreutzer, editors, *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[65] Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. Multilingual universal sentence encoder for semantic retrieval. In Asli Celikyilmaz and Tsung-Hsien Wen, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online, July 2020. Association for Computational Linguistics.

[66] Deepa Modi and Neeta Nain. Part-of-speech tagging of hindi corpus using rule-based method. In *Proceedings of the International Conference on Recent Cognizance in Wireless Communication & Image Processing: ICRCWIP-2014*, pages 241–247. Springer, 2016.

[67] Snigdha Paul, Mini Tandon, Nisheeth Joshi, and Iti Mathur. Design of a rule based hindi lemmatizer. In *Proceedings of Third International Workshop on Artificial Intelligence, Soft Computing and Applications, Chennai, India*, volume 2, pages 67–74, 2013.

[68] Arpana Prasad and Neeraj Sharma. Rule-based recognition of associated entities in hindi text: A domain centric approach. In *Information and Communication Technology for Competitive Strategies (ICTCS 2020) ICT: Applications and Social Interfaces*, pages 373–383. Springer, 2021.

[69] Deepakshi Singla and Parteek Kumar. Rule based anaphora resolution in hindi. In *2017 International Conference on Computational Intelligence in Data Science (ICCIDS)*, pages 1–5. IEEE, 2017.

[70] R. M. K. Sinha and A. Jain. AnglaHindi: an English to Hindi machine-aided translation system. In *Proceedings of Machine Translation Summit IX: System Presentations*, New Orleans, USA, September 23-27 2003.

[71] Sanjay K Dwivedi and Pramod P Sukhadeve. Translation rules for english to hindi machine translation system: homoeopathy domain. *Int. Arab J. Inf. Technol.*, 12(6A):791–796, 2015.

[72] Ariadna Font Llitjós, Jaime G Carbonell, and Alon Lavie. A framework for interactive and automatic refinement of transfer-based machine translation. In *Proceedings of the 10th EAMT Conference: Practical applications of machine translation*, 2005.

[73] Arafat Ahsan, Prasanth Kolachina, Sudheer Kolachina, Dipti Misra, and Rajeev Sangal. Coupling statistical machine translation with rule-based transfer and generation. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Research Papers*, Denver, Colorado, USA, October 31-November 4 2010. Association for Machine Translation in the Americas.

[74] Raj Nath Patel, Rohit Gupta, Prakash B. Pimpale, and Sasikumar M. Reordering rules for english-hindi smt, 2016.

[75] Omkar Dhariya, Shrikant Malviya, and Uma Shanker Tiwary. A hybrid approach for hindi-english machine translation. In *2017 International Conference on Information Networking (ICOIN)*, pages 389–394, 2017.

[76] Sanjay Chatterji, Devshri Roy, Sudeshna Sarkar, and Anupam Basu. A hybrid approach for bengali to hindi machine translation. In *Proceedings of ICON-2009: 7th international conference on natural language processing*, pages 81–91, 2009.

[77] Pramod Salunkhe, Mrunal Bewoor, and Suhas Patil. A research work on english to marathi hybrid translation system. *IJCSIT) International Journal of Computer Science and Information Technologies*, 6(3):2557–2560, 2015.

[78] Keerthi Lingam, E Ramalakshmi, and Srujana Inturi. English to telugu rule based machine translation system: A hybrid approach. *International Journal of Computer Applications*, 101(2), 2014.

[79] PG Anisree and KT Radhika. A hybrid translator: from malayalam to english. *Int. Res. J. Eng. Technol.(IRJET)*, 3(07):2395–0056, 2016.

[80] Seema Shukla. Framework for improving english to hindi rule-based translation system. *Linguistics International Journal*, 15(2):70–95, 2021.

[81] Shefali Saxena, Ayush Gupta, and Philemon Daniel. Efficient data augmentation via lexical matching for boosting performance on statistical machine translation for indic and a low-resource language. *Multimedia Tools and Applications*, pages 1–15, 2024.

[82] Abdullah Can Algan, Emre Yürekli, and Aykut Çayır. A use case: Reformulating query rewriting as a statistical machine translation problem. *arXiv preprint arXiv:2310.13031*, 2023.

[83] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[84] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122, 2017.

[85] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[86] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. volume 2, pages 1045–1048, 09 2010.

[87] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[88] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014.

[89] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[90] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[91] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[92] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

[93] Junbo Zhao Jiatao Gu, Changhan Wang. Levenshtein transformer. *Advances in Neural Information Processing Systems 32*, 2019.

[94] Amirhossein Tebbifakhr, Ruchit Agrawal, Matteo Negri, and Marco Turchi. Multi-source transformer with combined losses for automatic post editing. In Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 846–852, Belgium, Brussels, October 2018. Association for Computational Linguistics.

[95] Maja Popović. chrF: character n-gram F-score for automatic MT evaluation. In Ondřej Bojar, Rajan Chatterjee, Christian Federmann, Barry Haddow, Chris Hokamp, Matthias Huck, Varvara Logacheva, and Pavel Pecina, editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[96] Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. COMET: A neural framework for MT evaluation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online, November 2020. Association for Computational Linguistics.

[97] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling, 2019.

[98] Raunak Joshi and Abhishek Gupta. Performance comparison of simple transformer and res-cnn-bilstm for cyberbullying classification, 2022.

[99] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. Learning deep transformer models for machine translation, 2019.

[100] Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. Why self-attention? a targeted evaluation of neural machine translation architectures, 2018.

[101] Farhad Mortezapour Shiri, Thinagaran Perumal, Norwati Mustapha, and Raihani Mohamed. A comprehensive overview and comparative analysis on deep learning models: Cnn, rnn, lstm, gru, 2023.

[102] Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA, August 8-12 2006. Association for Machine Translation in the Americas.

[103] M. M. Schulreich, D. Breitschwerdt, J. Feige, and C. Dettbarn. Numerical studies on the link between radioisotopic signatures on earth and the formation of the local bubble: I. 60fe transport to the solar system by turbulent mixing of ejecta from nearby supernovae into a locally homogeneous interstellar medium. *Astronomy amp; Astrophysics*, 604:A81, August 2017.

[104] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, and Kenneth Heafield. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, 2018.

[105] Barret Zoph and Kevin Knight. Multi-source neural translation. In *Proceedings of NAACL-HLT 2016*, pages 30–34, 2016.

[106] Zhen-Hua Ling, Xiaojun Quan, Lei Li, Quan Liu, Xiaodan Zhu, and Zhen-Hua Ling. Co-attention based neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 191–201, 2018.

[107] Zhewen Yu, Jin Wang, Liang-Chih Yu, and Xuejie Zhang. Dual-encoder transformers with cross-modal alignment for multimodal aspect-based sentiment analysis. In Yulan He, Heng Ji, Sujian Li, Yang Liu, and Chua-Hui Chang, editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 414–423, Online only, November 2022. Association for Computational Linguistics.

[108] Jindřich Libovický, Jindřich Helcl, and David Mareček. Input combination strategies for multi-source transformer decoder, 2018.

[109] Kätriin Kukk. Automatic post-editing and quality estimation in machine translation of product descriptions. Master's thesis, Uppsala University, Department of Linguistics and Philology, 2022.

[110] Pushpak Bhattacharyya, Rajen Chatterjee, Markus Freitag, Diptesh Kanojia, Matteo Negri, and Marco Turchi. Findings of the WMT 2022 shared task on automatic post-editing. In Philipp Koehn, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee,

Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 109–117, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics.

[111] Pushpak Bhattacharyya, Rajen Chatterjee, Markus Freitag, Diptesh Kanojia, Matteo Negri, and Marco Turchi. Findings of the WMT 2023 shared task on automatic post-editing. In Philipp Koehn, Barry Haddow, Tom Kocmi, and Christof Monz, editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 672–681, Singapore, December 2023. Association for Computational Linguistics.

[112] Xu Zhang and Xiaojun Wan. An empirical study of automatic post-editing, 2022.

[113] Bohan Li, Yutai Hou, and Wanxiang Che. Data augmentation approaches in natural language processing: A survey. *AI Open*, 3:71–90, 2022.

[114] Hyeonseok Moon, Chanjun Park, Sugyeong Eo, Jaehyung Seo, SeungJun Lee, and Heuiseok Lim. A self-supervised automatic post-editing data generation tool, 2022.

[115] Multidimensional Quality Metrics (MQM). Mqm definition. `http://www.qt21.eu/mqm-definition/`. Accessed: 2024-05-15.

[116] Pushpak Bhattacharyya, Rajen Chatterjee, Markus Freitag, Diptesh Kanojia, Matteo Negri, and Marco Turchi. Findings of the WMT 2022 shared task on automatic post-editing. In Philipp Koehn, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 109–117, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics.

[117] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[118] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[119] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[120] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[121] Tao Ge, Furu Wei, and Ming Zhou. Fluency boost learning and inference for neural grammatical error correction. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1055–1065, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[122] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[123] Vikas Raunak, Amr Sharaf, Yiren Wang, Hany Hassan Awadallah, and Arul Menezes. Leveraging gpt-4 for automatic translation post-editing, 2023.

[124] Yasmin Moslem, Gianfranco Romani, Mahdi Molaei, Rejwanul Haque, John D. Kelleher, and Andy Way. Domain terminology integration into machine translation: Leveraging large language models, 2023.

[125] Yasmin Moslem, Gianfranco Romani, Mahdi Molaei, Rejwanul Haque, John D. Kelleher, and Andy Way. Domain terminology integration into machine translation: Leveraging large language models, 2023.

[126] Serge Gladkoff, Gleb Erofeev, Irina Sorokina, Lifeng Han, and Goran Nenadic. Predictive data analytics with ai: assessing the need for post-editing of mt output by fine-tuning openai llms, 2023.

[127] Blanca Vidal, Albert Llorens, and Juan Alonso. Automatic post-editing of MT output using large language models. In Janice Campbell, Stephen Larocca, Jay Marciano, Konstantin Savenkov, and Alex Yanishevsky, editors, *Proceedings of the 15th Biennial Conference of the Association for Machine Translation in the Americas (Volume 2: Users and Providers Track and Government Track)*, pages 84–106, Orlando, USA, September 2022. Association for Machine Translation in the Americas.

[128] Amr Hendy, Mohamed Abdelrehim, Amr Sharaf, Vikas Raunak, Mohamed Gabr, Hitokazu Matsushita, Young Jin Kim, Mohamed Afify, and Hany Hassan Awadalla. How good are gpt models at machine translation? a comprehensive evaluation, 2023.

[129] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets.

[130] Jindřich Libovický, Jindřich Helcl, Marek Tlustý, Ondřej Bojar, and Pavel Pecina. Cuni system for wmt16 automatic post-editing and multimodal translation tasks. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 646–654, Berlin, Germany, August 2016. Association for Computational Linguistics.

[131] Ryoya Yuasa, Akihiro Tamura, Tomoyuki Kajiwara, Takashi Ninomiya, and Tsuneo Kato. Multimodal neural machine translation using synthetic images transformed by latent diffusion model. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 76–82, Toronto, Canada, 2023. Association for Computational Linguistics.