Chapter 3

Common Phonetic-Orthographic Representation

3.1 Introduction

Statistical Machine Translation *(SMT)* and Neural Machine Translation *(NMT)* are two most widely used architectures for MT. Unlike traditional MT systems [73, 74], SMT is a log-linear framework consisting of language and translation models [75], whereas NMT is an end-to-end neural network-based encoder-decoder model. Both of them predict the likelihood of a sequence of words in the target language being a correct translation of the source sentence either using a purely probabilistic approach (SMT) or a deterministic/probabilistic weight learning approach (many kinds of NMT). The encoders generate context vectors for input sentences and decoders decode these vectors to generate target sequences. [76] introduced an attention mechanism in encoder for putting more weights on the words that contained better context vectors of sentences. Using on the attention mechanism, many improvements have been introduced in NMT using Deep Learning, such as Transformer [14], BART [77] and mBART [28] in the recent years.

Both of the above kinds of MT models require a huge parallel corpus. NMT has

achieved success in dealing with the need of huge parallel resources by introducing various techniques such as back-translation [78], domain adaptation [79], and fine-tuning [80]. NMT provides new opportunities for better translation for High Resource Languages *(HRLs)* and to some extent for Low Resource Languages *(LRLs)*. Here, HRLs are the language pairs for which parallel corpus is available in huge amounts to train the model (e.g., German↔English and French↔English). In comparison, LRLs are the language pairs in which training data is insufficient for good enough SMT or NMT, e.g., Nepali↔Hindi, Marathi↔Hindi. Insufficient training data works as an obstacle for NMT in improving the translation quality for LRLs, due to things like missing context and rare word problems. Some techniques introduced by Sennrich et al. [68] and Fei et al. [57] have been tried to address such issues.

Training an effective and accurate MT system requires a large amount of parallel corpus consisting of source and target language pairs. When we talk about low-resource languages, the first problem is to find a fair amount of parallel corpus, which makes it challenging to create tools and applications for extremely poor-resource languages. Creating a large parallel corpus for MT for each language pair that falls into the low resource category is an expensive, time-consuming, and labour-intensive task. So, the solution to improve NMT in low resource context is by leveraging the morphological, structural, functional, and perhaps deep semantic features of such languages. Fortunately, for similar languages, it is possible to exploit the similarities for better modelling of closely related languages. For this, we need to focus on features that help the MT system better learn the close relationship between such languages. Conference on Machine Translation (WMT) has also started shared tasks for similar language translations from 2019 [81].

When we talk about Indian languages, most languages except Hindi come under extremely low resource categories. Even Hindi is, from some points of view either a low or medium resource language. India being a country with rich linguistic diversity,

there is a need for MT systems across the Indian (or South Asian) languages. India is also inhabited by a vast population who speak languages belonging to three prominent families, Indo-Aryan (a subfamily of Indo-European), Dravidian, and Tibeto-Burman, but due to very long contact and interactions, they have gone through a process of 'convergence', forming India as a linguistic area [10].

For some of the major languages, and even for some of the 'regional' or 'minority languages' (since they were widely used for a long duration in the past for literary purposes), there are records available and there is a varying degree of the well-developed tradition of at least (spoken) literary usage. However, only some languages, which are officially recognized, have some written tradition, particularly for non-literary prose. The rest have very little written data, or even if it is there, it is usually not in a machine-readable format. Therefore, they can be treated as extremely low or zero-resource languages. There is a need for the development of MT systems for such languages, and the similarity between the languages helps in developing such MT systems.

In this chapter, we propose an approach based on leveraging the features of similar languages by simply, programmatically[1], converting them into an intermediate Latin-based multilingual notation. The notation that we use here is the commonly used WX-notation [6], which is often used in NLP tools and systems for Indian languages developed in India. This notation (like many other similar notations) can project all the Indic or Brahmi origin scripts [7], which have — in many cases — different Unicode blocks, into a common character space. Our intuition is that this should help in capturing phonological, orthographic, and, to some extent, morphosyntactic similarities that will help a neural network-based model in better multilingual learning and translation across this languages [4, 5, 9]. We do this by using this WX-converted text to learn byte pair encoding-based embeddings. The effect of this is that the similar but different languages are projected onto the same orthographic-phonetic space [8], and hence

[1]Using encoding converters, such as `https://pypi.org/project/wxconv/`

also in the same common morphological and lexical space, allowing better modeling of multilingual relationships in the context of India as a linguistic area.

In addition, using WX has another benefit, even for a single script such as Devanagari. Brahmi-derived scripts have different symbols for dependent vowels (called *maatraas*) which modify a consonant and independent vowels (written as *aksharas*) which are pronounced as syllables. WX uses the same symbols for these two variants of the same vowel, while Unicode uses different codes and the scripts themselves use different graphical symbols.

After conversion to WX, we apply some of the state-of-the-art NMT techniques to build our MT systems. These NMT systems, such as the Transformer, should learn better the relationships between languages.

Table 3.1: Some details about the languages used in our experiments

Languages	Family	Script	Word Order	Ergative	Place
Hindi		Devanagari		Yes	Mainly North India
Gujarati		Gujarati		No	Mainly Gujarat
Marathi		Balbodh version of Devanagari		No	Mainly Maharashtra
Nepali	Indo-Aryan	Devanagari	SOV	Yes	Mainly Nepal
Maithili		Devanagari		No	Mainly Bihar and parts of Nepal
Punjabi		Gurumukhi		No	Mainly Punjab
Urdu		Variant of Perso-Arabic		No	Mainly North India

We select six pairs of similar languages: Gujarati (GU)↔Hindi (HI), Marathi (MR)↔Hindi (HI), Nepali (NE)↔Hindi (HI), Maithili (MAI)↔Hindi (HI), Punjabi (PA)↔Hindi (HI), and Urdu (UR)↔Hindi (HI). Table 3.1 contains some of the language features that help in figuring out how selected languages are similar to Hindi. For example, Hindi, Gujarati, Marathi, Nepali, Maithili, Punjabi, and Urdu belong to the Indo-Aryan Language family, and all the selected languages except Punjabi, Gujarati and Urdu share a common Devanagari script. Even Gujarati and Punjabi scripts are derived from the ancient Brahmi script, so they have the same set of abstract symbols, except that they use different graphical symbols to represent them. The word order of all the selected languages is mostly $Subject + Object + Verb$. Apart from this, all these languages share lexical similarities with Hindi in terms of common words derived

from Sanskrit and other languages as mentioned earlier. Also, these languages have phonological similarities with Hindi.

The contributions of this chapter are summarized as follows:

1. Propose a WX-based machine translation approach that leverages orthographic and phonological similarities between pairs of Indian languages.
2. Proposed approach achieves an improvement of *+0.01* to *+10* BLEU points compared to baseline state-of-the-art techniques for similar language pairs in most cases. We also get *+1* BLEU points improvement on distant and zero-shot language pairs.

3.2 Proposed Approach

To tackle the morphological richness related problems in NMT training for Indian languages and to be able work with very little resources, we propose a simple but effective approach for translating low-resource languages that are similar in features and behaviour.

The proposed approach consists of three modules: Text Encoder, Model Trainer, and Text Decoder (Figure 3.1), as discussed in the following section.

3.2.1 Text Encoder

The proposed model first encodes the source and target corpora of parallel languages into an intermediate representation, the WX-notation[2] [82]. The primary reason behind encoding the source and target language corpora into WX-notation is to encode different languages with the same or different scripts into a common representation by projecting them onto a common phonetic-orthographic character space so that BPE can be linguistically better informed. WX-notation is a transliteration scheme for representing Indian languages in ASCII format, and as described earlier, it has many advantages as

[2] `https://pypi.org/project/wxconv/`

an intermediate representation, even compared to using Devaganari or any other single Brahmi-based script. It implicitly helps the Transformer encoder model more cognates, loan words, and morphologically similar words between the languages, as well as model other kinds of similarities for better translation.

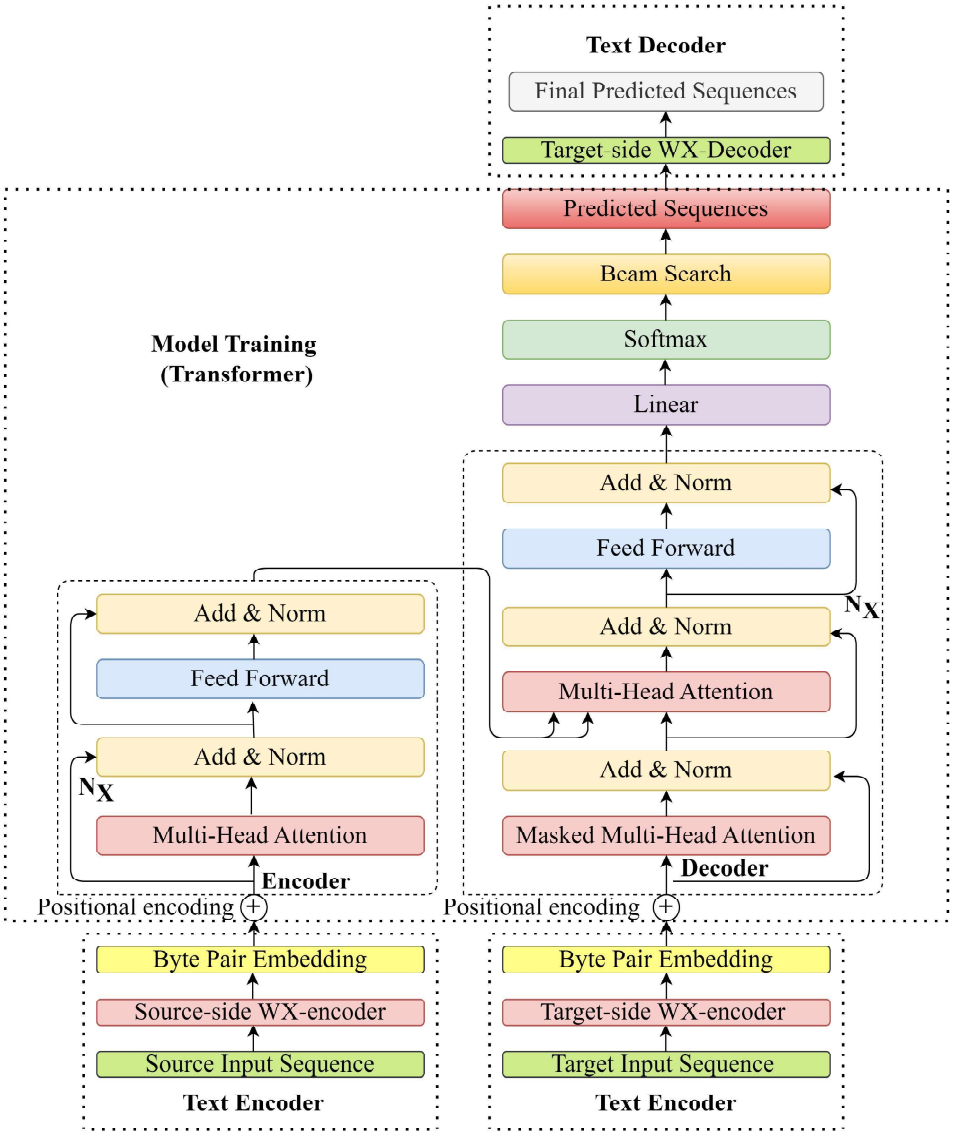

Figure 3.1: Proposed architecture.

3.2.2 Model Training

The intermediate representation of the source language text is passed to the Transformer encoder. The Transformer encoder-decoder model learns the relationship between languages. We have used the SentencePiece[3] library for tokenization of the text. SentencePiece is used as a pre-processing task for the WX-encoded source-target text in the concerned language pair. SentencePiece is a language-independent sub-word tokenizer and detokenizer designed for Neural-based text processing, including neural machine translation. It implements two subword segmentation algorithms, Byte-Pair Encoding (BPE) and unigram language model, with direct training from raw sentences [68, 83]. Therefore, it already indirectly, to some extent, provides cognates, loan words, and morphologically similar words to the Transformer, and our prior conversion to WX allows it to do so better. It may be noted that the approach is generalizable to other multilingual transliteration notations, perhaps even to IPA[4,5], which is almost truly phonetic notation for written text.

3.2.3 Text Decoder

After convergence of the training algorithm, the WX-encoded generated target sentences are decoded back to the plain text format to evaluate the model.

3.3 Corpus and Experimental Settings

In this section, we discuss the corpus statistics and experimental settings required to perform the experiment.

[3]https://github.com/google/sentencepiece
[4]`https://en.wikipedia.org/wiki/International_Phonetic_Alphabet_chart`
[5]`https://www.internationalphoneticassociation.org/`

Table 3.2: Corpus Statistics showing the number of training, validation, and test sentences for each domain

Lang-Pairs	Train	Validation	Test	Domain
GU↔HI	14784	1000	1973	PM India
NE↔HI	133991	3000	3000	WMT 2019 corpus, Agriculture, Entertainment, Bible
MR↔HI	42274	1000	1411	News, PM India, Indic WordNet
PA↔HI	225576	7199	7200	GNOME, KDE4, Ubuntu, wikimedia, TED2020
MAI↔HI	93136	2972	2973	GNOME, KDE4, wikimedia, Ubuntu
UR↔HI	108176	3452	3453	Tanzil, GNOME, KDE4, wikimedia, Ubuntu
ML↔HI	16833	500	500	PM India
TA↔HI	43038	500	500	PM India
TE↔HI	2584	500	500	PM India
BHO↔HI	0	500	500	Movie subtitles, Literature, News
MAG↔HI	0	500	500	Movie subtitles, Literature, News

Note- HI: Hindi, MR: Marathi, NE: Nepali, GU: Gujarati, MAI: Maithili, PA: Punjabi, UR: Urdu, ML: Malayalam, TA: Tamil, TE: Telgu, BHO: Bhojpuri, MAG: Magahi

3.3.1 Corpus description

We evaluate the proposed model in an extremely low-resource scenario on the mutually similar languages which we selected for our experiments. These are Hindi (HI), Gujarati (GU), Marathi (MR), Nepali (NE), Maithili (MAI), Punjabi (PA), Urdu (UR), Bhojpuri (BHO), Magahi (MAG), Malayalam (ML), Tamil (TA) and Telgu (TE). We perform experiments on the following language pairs involving Hindi: GU↔HI, NE↔HI, MR↔HI, MAI↔HI, PA↔HI, and UR↔HI. Parallel corpora of GU↔HI, ML↔HI, TA↔HI, and TE↔HI for training, testing, and validation are downloaded from CVIT-PIB [84]. MR↔HI parallel corpus is collected from WMT 2020 shared tasks[6]. NE↔HI language pair corpus is made up of those collected from WMT 2019 shared tasks [7], Opus [8], and TDIL [9] repositories. We use a monolingual corpus of Gujarati, Hindi, and Marathi for similarity computation in section 5.1 from the PM India dataset described in [85]. The rest of the monolingual corpora are collected from the Opus collection for similarity computation in section 5.1 [86]. We use SentencePiece [69] to pre-process the source and target sentences. We use 5K merge operations to learn BPE with the

[6] `http://www.statmt.org/wmt20/similar.html`
[7] http://www.statmt.org/wmt19/index.html
[8] https://opus.nlpl.eu/
[9] http://www.tdil-dc.in/index.php?lang=en

SentencePiece model and restrict the source and target vocabularies to at most 5K tokens. There are some places where code-switching occurs in the employed dataset. The WX-transliteration tool ignores code-switched data and keeps it in the datasets as it is.

Table 3.3: Experiment results (BLEU, chrF2, and TER scores).

Languages(xx)	BLEU		chrF2		TER	
	XX→HI					
	Guzmán et.al [58]	Proposed	Guzmán et.al [58]	Proposed	Guzmán et.al [58]	Proposed
GU	33.14	**33.15**	**58**	57	**0.541**	0.548
NE	30.51	**41.97**	46	**49**	0.658	**0.652**
MR	16.87	**22.37**	43	**44**	**0.707**	0.709
PA	78.56	**81.05**	82	**82**	0.220	**0.216**
UR	28.74	**30.08**	45	**45**	0.668	**0.657**
MAI	79.49	**81.80**	**82**	81	**0.242**	0.251
	HI→XX					
	Guzmán et.al [58]	Proposed	Guzmán et.al [58]	Proposed	Guzmán et.al [58]	Proposed
GU	25.47	**25.82**	56	**56**	**0.616**	0.619
NE	32.89	**43.52**	50	**51**	**0.630**	0.637
MR	14.05	**14.76**	41	**44**	0.789	**0.762**
PA	80.01	**81.87**	83	**84**	0.206	**0.203**
UR	22.74	**24.35**	46	**47**	0.597	**0.596**
MAI	**86.58**	83.82	**89**	86	**0.148**	0.168

3.3.2 Training details

3.3.2.1 Proposed approach

We use the WX conversion tool [10] for transliterating the text and the fairseq [11] [87] toolkit, which is a sequence modelling toolkit, to train the Transformer. We use five encoder and decoder layers. The encoder and decoder embedding dimensions are set to 512. Feed-forward encoding and decoding embedding dimensions are set to 2048. The number of an encoder and decoder attention heads is set to 2. The dropout, the attention dropout, and the ReLU dropout are set to 0.4, 0.2, and 0.2, respectively. The weight decay is set at 0.0001, and the label smoothing is set to 0.2. We use the Adam optimizer, with β_1 and β_2 set to 0.9 and 0.98. The learning rate schedule is inverse square root, with an initial learning rate of 1e-3 and a minimum learning rate of 1e-9. The maximum number of tokens used is set to 4000. The maximum number of epochs

[10] `https://pypi.org/project/wxconv/`
[11] https://github.com/facebookresearch/fairseq

for training is set to 100. We use a beam size equal to 5 for generating data using the test set.

3.3.2.2 Guzmán et al. [58]

In Guzmán et al. [58], authors have demonstrated the experiments on extremely low resource languages using Transformer. Our proposed approach is based on the Transformer described in Guzmán et al. [58] with the addition of two extra modules, Text Encoder and Text Decoder. We use the Transformer model described in Guzmán et al. [58] as a reasonably high baseline to compare the proposed approach without the intermediate representation of the WX-notation for Indian languages. The projection to WX could be used for any other NMT approach as well that uses a subword embedding.

3.3.2.3 SMT

We use Moses [12], an open-source toolkit to train SMT [88]. For obtaining the phrase/word alignments from parallel corpora, we use *GIZA++* [89]. A 5-gram KenLM language model is used for training [90]. The parameters are tuned on the validation set using *MERT* and tested with a test set [91].

3.4 Results and Analysis

We compare the proposed approach with the Moses-based SMT and the Transformer-based NMT model [58], where the latter is used as the baseline for NMT. We use three evaluation metrics, BLEU [13] [72], TER [92] and chrF2 [93] for better comparison of the proposed approach. We see from Table 3.3 that the proposed approach improves upon the baseline for most of the pairs.

We also perform a comparison between SMT without WX-transliteration and SMT

[12] http://www2.statmt.org/moses/
[13] `https://github.com/mjpost/sacrebleu`

Table 3.4: BLEU score-based comparison of SMT, SMT + WX and the proposed approaches.

Languages(xx)	BLEU		
	XX→HI		
	SMT	**SMT + WX**	**Proposed**
GU	**43.49**	30.69	33.15
NE	40.14	**53.21**	41.97
MR	**7.41**	1.46	22.37
PA	68.34	**71.22**	81.05
UR	19.21	**21.84**	30.08
MAI	79.56	**81.46**	81.80
	HI→XX		
	SMT	**SMT + WX**	**Proposed**
GU	**39.20**	25.89	25.82
NE	40.21	**54.84**	43.52
MR	**7.36**	1.48	14.76
PA	67.21	**70.64**	81.87
UR	18.24	**18.41**	24.35
MAI	79.12	**83.06**	83.82

with it. These two sets of results are also compared with the proposed approach as shown in Table 3.4. In the case of SMT also we can easily note that the performance improves in most cases by using WX as the intermediate notation, even though SMT is not using subword embeddings.

We also present some basic analysis of the scores as shown in Table 3.3. We use corpus-based language relatedness and complexity measures for further analysis for this purpose in the next section.

Table 3.5: Similarity between languages using SSNGLMScore

Model	BHO	GU	HI	MAG	MAI	ML	MR	NE	PA	TA	TE	UR
BHO	-	0.5659	0.6725	0.6997	0.7235	0.4090	0.5687	0.4979	0.4580	0.3233	0.5057	0.4237
GU	-	-	0.5483	0.5642	0.6449	0.3727	0.5411	0.3868	0.3408	0.2531	0.4578	0.3787
HI	-	-	-	0.6331	0.6598	0.3536	0.5717	0.4181	0.4046	0.2564	0.4567	0.3670
MAG	-	-	-	-	0.7762	0.4414	0.5724	0.5671	0.4827	0.3736	0.5248	0.5245
MAI	-	-	-	-	-	0.5833	0.6496	0.6968	0.5734	0.5453	0.6435	0.7040
ML	-	-	-	-	-	-	0.3736	0.3388	0.1968	0.3792	0.4507	0.2759
MR	-	-	-	-	-	-	-	0.4023	0.3496	0.2637	0.4771	0.3498
NE	-	-	-	-	-	-	-	-	0.2661	0.2784	0.3985	0.4354
PA	-	-	-	-	-	-	-	-	-	0.1449	0.2718	0.2938
TA	-	-	-	-	-	-	-	-	-	-	0.2972	0.2641
TE	-	-	-	-	-	-	-	-	-	-	-	0.3493
UR	-	-	-	-	-	-	-	-	-	-	-	-

3.4.1 Similarity between languages

Since there are no definitive methods to judge the similarity between two languages, we use the following techniques to compute the similarity between the languages:

3.4.1.1 SSNGLMScore

We use character-level n-gram language models based SSNGLMScore to measure the relatedness between languages [94, 95]. SSNGLMScore is computed as follows:

$$S_{sl,tl} = \sum_{tl=1}^{m} p_{sl.tl}(w_n|w_1^{n-1}), \tag{3.1}$$

where S stands for Scaled Sum of n-gram language model scores.

$$MS_{sl,tl} = \frac{S_{sl,tl} - \min(S_{SL,TL})}{\max(S_{SL,TL}) - \min(S_{SL,TL})}, \tag{3.2}$$

where, sl and tl represent the source language and the target language, respectively. Moreover, $sl \in$ SL(Gujarati, Marathi, Maithili, Nepali, Urdu, Punjabi, Hindi, Malayalam, Tamil, Telugu, Bhojpuri, Magahi) and m is the total number of sentences in the target language $tl \in$ TL(Gujarati, Marathi, Maithili, Nepali, Urdu, Punjabi, Hindi, Malayalam, Tamil, Telugu, Bhojpuri, Magahi). We train the language model using a 6-gram character-level KenLM model on the source monolingual corpus (sl). Each language model is tested on target language (tl), and the scores are reported.

Table 3.5 lists the cross-lingual similarity scores of Hindi, Gujarati, Marathi, Nepali, Maithili, Punjabi, Malayalam, Tamil, Telugu, Bhojpuri, Magahi, and Urdu with each other. Based on SSNGLMScore, Bhojpuri, Maithili and Magahi are the closest to Hindi, which matches linguistic knowledge about them, whereas Urdu seems to as far from Hindi as Malayalam and more than Telugu. The reasons Urdu is far from Hindi is partly that Urdu is written in a different kind of script from Hindi which does not have a straightforward mapping to WX, but mainly because, though grammatically almost

identical, the two use very different vocabularies in written and formal forms. Maithili is also the second official language of Nepal and is also highly similar to Nepali, perhaps due to prolonged close contact. What is more surprising is that the similarity between Urdu and Nepali is relatively high, whereas that between Urdu and Hindi is among the lowest. This could be because of the nature of the corpus. Going through Tables 3.3, we find that there is an improvement in every metric except TER in a majority of cases when we apply the proposed method on the translation direction from Maithili, Gujarati, Marathi, Nepali, Punjabi, and Urdu to Hindi. This observation allows us to assert that the proposed approach improves performance for translation between similar languages. Thus, even though the similarity measure we used mixes different kinds of similarities, it is suitable for our purposes because our method is based on sub-word and multilingual modelling.

We also see a gain of +1.34 BLEU points on Hindi to Urdu despite Urdu being far away from the rest of the language pairs in terms of the similarity score we used. There is a considerable improvement of +11.46 BLEU points on HI→NE and +10.63 BLEU points on NE→HI language pairs.

Table 3.6: char-BLEU score on the training data

Languages	**char-BLEU**
Gujarati↔Hindi	47.29
Marathi↔Hindi	35.05
Nepali↔Hindi	40.53
Maithili↔Hindi	66.70
Punjabi↔Hindi	37.17
Urdu↔Hindi	8.61

Note- Applying char-BLEU score on the training data of both the languages of the pair

Table 3.7: TER and chrF2 scores on the training data

Languages	$GU \rightarrow HI$	$MR \rightarrow HI$	$NE \rightarrow HI$	$MAI \rightarrow HI$	$PA \rightarrow HI$	$UR \rightarrow HI$
TER	1.066	1.300	1.052	0.610	0.988	1.093
chrF2	38	29	34	65	32	12
Languages	$HI \rightarrow GU$	$HI \rightarrow MR$	$HI \rightarrow NE$	$HI \rightarrow MAI$	$HI \rightarrow PA$	$HI \rightarrow UR$
TER	0.884	0.940	0.887	0.555	0.906	1.044
chrF2	39	29	36	62	30	10

Note- Applying TER and chrF2 scores on the training data of both the languages of the pair

3.4.1.2 char-BLEU, TER and chrF2

To better understand the slight fall in BLEU points despite the similarity for MAI → HI and large increment in the case of NE↔Hi (where Nepali and Maithili are known to be close), we also compute similarity by applying char-BLEU [96], chrF2, and TER on a training dataset of all language pairs. The reason behind using char-BLEU and chrF2 for similarity is that since they are character-based metrics, there is a greater chance of covering the morphological aspects. Before calculating the char-BLEU, the TER, and the chrF2 evaluation metrics, data must be in the same script to evaluate the score. So, we convert the corpus from UTF-8 to WX-notation. Table 3.6 contains the char-BLEU score of language pairs, whereas Table 3.7 contains the TER and chrF2 scores of each language pair. We see Table 3.6 and 3.7 and find out that HI and MAI are still more similar compared to other pairs. We can only hypothesize the reason being that this is due to the nature of the data that we have used.

3.4.2 Analysis of language complexity

3.4.2.1 Morphological complexity

Since Indian languages are morphologically rich, machine translation systems based on word tokens have difficulty with them. Therefore, we also tried to relate the results obtained with estimates of such complexity obtained from character-level entropy. It is reasonable to assume that the greater the character-level entropy, the more morphologically complex a language is likely to be.

Character-level entropy We used Character-level word entropy to estimate morphological redundancy, following Bharati et al. [97] and Bentz and Alikaniotis 2016 [98].

A "word" is defined in our experiments as a space-separated token, i.e., a string of alphanumeric Unicode characters delimited by white spaces. The average information content of character types for words is then calculated in terms of Shannon entropy [99]:

$$H(T) = -\sum_{i=1}^{V} p(c_i) \log_2(p(c_i)) \tag{3.3}$$

Table 3.8 lists the word (unigram) entropy of languages at character level, which indirectly represents languages' lexical richness, i.e., how complex – in terms of characters they are made up of – word forms are. Since we compute the unigram entropy based on characters, we can say that lexical richness also indicates morphological complexity, both derivational and inflectional. Based on the corpus-based word entropy values, it appears that Hindi is more morphologically complex than the other six languages. However, this may be more of derivational complexity rather than inflectional complexity, as Hindi is relatively simpler in terms of inflectional morphology. The high derivational complexity of Hindi is because it is the official language of India and is more standardized than most other Indian languages. It, therefore, has borrowed and coined a large number of complicated words and technical terms, whether from Persian or Sanskrit or English. This adds a great deal to the derivational complexity of written formal Hindi, compared to commonly spoken Hindi. At least, this is our hypothesis based on the similarity and complexity results.

We also find that our approach shows a considerable improvement of about more than 10 BLEU points in both directions for the Hindi-Nepali language pair, i.e., NE→HI and HI→NE. Such improvement may be attributed to the effect caused by projecting to a common multilingual orthographic-phonetic notation, that is, WX. This probably helps the Transformer learn the context between languages better with the help of a sentence piece tokenizer.

In Tables 3.9, 3.10 and 3.11, we present the values of word entropy and redundancy at character level. These tables show that the entropy increases when converting to WX and redundancy decreases. This is evidence of the fact that the projection to a common orthrographic and phonetic space causes the entropy to increase and redundancy to decrease, thus allowing more compact representations to be learnt from the data after conversion to WX in our case.

Table 3.8: Character-based entropy of languages with or without applying WX-notation

Languages	**Character Entropy**	**Character Entropy***	Difference
Gujarati	5.0368	3.7454	1.2914
Marathi	5.0220	3.6846	1.3374
Nepali	4.6722	3.5770	1.0952
Maithili	5.1159	3.9162	1.1997
Punjabi	5.0834	3.7932	1.2902
Urdu	4.8821	4.1198	0.7623
Hindi	5.2195	3.7974	1.4221

* After applying WX-notation

Table 3.9: Character-level Entropy computed on Monolingual Vocabulary

Language	Complete corpus						Restricted corpus					
	Without WX			With WX			Without WX			With WX		
	Max	Median	Average	Max	Median	Average	Max	Median	Average	Max	Median	Average
HI	3.1674	0.5897	0.6196	4.9433	1.2484	1.3148	3.1623	0.5929	0.6230	4.9414	1.2495	1.3158
GU	6.4712	0.8113	0.8389	17.9337	1.4677	1.5157	6.4735	0.8128	0.8410	22.2253	1.4681	1.5163
NE	3.0311	0.8008	0.8287	6.6845	1.4327	1.4835	1.8080	0.5350	0.5636	4.7487	1.1262	1.1575
MR	3.7534	0.5982	0.6281	7.7372	1.2331	1.2995	3.5845	0.8049	0.8459	7.7400	1.2130	1.2734
PA	2.2077	0.5778	0.6048	8.9978	1.0349	1.1105	2.1662	0.5500	0.5753	13.5759	0.9644	1.0405
UR	2.8580	0.6484	0.6786	3.092	0.7748	0.8088	2.2477	0.6282	0.6574	3.3297	0.7523	0.7828
MAI	2.0163	0.5097	0.5326	4.3135	1.0904	1.1432	1.6417	0.4773	0.5003	3.8923	1.0401	1.0888

Table 3.10: Character-level Redundancy Reduction on Monolingual Corpus

Languages	Complete corpus		Restricted corpus	
	Without WX	WX	Without WX	WX
HI	0.8955	0.7693	0.8949	0.7691
GU	0.8606	0.7401	0.8603	0.7400
NE	0.8806	0.7866	0.9111	0.8147
MR	0.9050	0.7993	0.8610	0.7807
PA	0.9186	0.8502	0.9194	0.8554
UR	0.8941	0.8741	0.8968	0.8750
MAI	0.9125	0.8121	0.9172	0.8171

Table 3.11: Character-level Entropy and Redundancy Changes for Parallel Corpus

Language pair	Without WX				With WX			
	Maximum Entropy	Median Entropy	Average Entropy	Redundancy	Maximum Entropy	Median Entropy	Average Entropy	Redundancy
GU-HI	4.8292	0.43224	0.4985	0.9279	17.7731	1.3958	1.4509	0.7512
NE-HI	3.0273	0.7414	0.7725	0.8948	7.1454	1.3561	1.4126	0.7988
MR-HI	3.7557	0.6003	0.6303	0.9047	7.7342	1.2309	1.2977	0.7995
PA-HI	1.6642	0.3359	0.3510	0.9543	9.0232	1.1199	1.1843	0.8414
UR-HI	1.9841	0.3547	0.3864	0.9489	4.0133	0.7928	0.8472	0.8783
MAI-HI	2.0483	0.5340	0.5555	0.9096	6.8270	1.1097	1.1656	0.8091

3.4.2.2 Syntactic complexity

Perplexity Perplexity (PP) of a language can be seen as a weighted average of the reciprocal of its branching factor [94]. Branching factor is the number of possible words that can succeed any given word based on the context. Therefore, perplexity – as a kind of mean branching factor – is a mean representative of the possible succeeding words given a word. Thus, it can be seen as a rough measure of the syntactic complexity. If the model is a good enough representation of the true distribution for the language, then the PP value will actually indicate syntactic complexity.

To estimate distances of other languages from Hindi using perplexity, we trained the perplexity model on the Hindi corpus and tested it on the corpora of other languages.

$$PP(C) = \sqrt[W]{\frac{1}{P(S_1, S_2, S_3, ..., S_n)}} \tag{3.4}$$

where corpus C contains n sentences with W words.

Table 3.12 and 3.13 contain the assymmetric and symetric perplexity — average of the two translation directions — values between the concerned language pairs and indicate their distances from Hindi based on character-level language model. Pairs having higher perplexity scores means the languages are more distant. We see language pairs Urdu and Hindi have more perplexity scores. This is mostly because these two languages, though almost identical in spoken form and in terms of core syntax and core vocabulary, use very different extended vocabularies for written and formal purposes,

Table 3.12: Cross-lingual distance between languages after applying character-level language model using perplexity-based score (Unnormalized on language directions)

Language	BHO	GU	HI	MAG	MAI	ML	MR	NE	PA	TA	TE	UR
BHO	0.0010	0.0443	0.0280	0.0290	0.0617	0.1006	0.0418	0.1648	0.0507	0.1383	0.0790	0.3134
GU	0.0319	0.0	0.0312	0.0504	0.0704	0.0648	0.0302	0.1736	0.0663	0.1117	0.0556	0.2675
HI	0.0116	0.0312	0.0007	0.0290	0.0715	0.0900	0.0190	0.1670	0.0458	0.1393	0.0705	0.2933
MAG	0.0414	0.0992	0.0712	6.3465e-06	0.0739	0.1897	0.0924	0.1710	0.0834	0.2036	0.1693	0.3491
MAI	0.0806	0.0875	0.0891	0.1340	0.0002	0.1394	0.0986	0.1769	0.0941	0.2168	0.1295	0.4006
ML	0.0713	0.0667	0.0773	0.0962	0.0790	0.0002	0.0695	0.1323	0.1171	0.0497	0.0403	0.3785
MR	0.0308	0.0280	0.0314	0.0503	0.0682	0.0623	0.0007	0.1625	0.0644	0.1175	0.0445	0.3423
NE	0.0949	0.1536	0.1370	0.1065	0.0955	0.1962	0.1321	0.0003	0.2130	0.2506	0.1862	0.3350
PA	0.0545	0.0935	0.0612	0.0782	0.0892	0.1573	0.0785	0.2762	0.0003	0.1716	0.1485	0.3245
TA	0.1239	0.1439	0.1384	0.1595	0.1009	0.0487	0.1204	0.1761	0.1613	0.0003	0.0972	0.3910
TE	0.0511	0.0539	0.0562	0.0785	0.0783	0.0449	0.0510	0.1513	0.1102	0.1165	0.0002	0.3401
UR	1.0	0.2823	0.5221	0.4771	0.1984	0.4330	0.4014	0.6438	0.3150	0.3276	0.5548	0.0001

Table 3.13: Cross-lingual distance between languages after applying character-level language model using perplexity-based score

Languages	BHO	GU	HI	MAG	MAI	ML	MR	NE	PA	TA	TE	UR
BHO	0.0	0.0381	0.0198	0.0352	0.0712	0.0860	0.0363	0.1298	0.0526	0.1311	0.0650	0.6567
GU	-	0.0	0.0312	0.0748	0.0789	0.0658	0.0291	0.1636	0.0799	0.1278	0.0548	0.2749
HI	-	-	0.0	0.0501	0.0803	0.0836	0.0252	0.1520	0.0535	0.1388	0.0634	0.4077
MAG	-	-	-	0.0	0.1040	0.1430	0.0713	0.1387	0.0808	0.1815	0.1239	0.4131
MAI	-	-	-	-	0.0	0.1092	0.0834	0.1362	0.0916	0.1589	0.1039	0.2995
ML	-	-	-	-	-	0.0	0.0659	0.1642	0.1372	0.0492	0.0426	0.4057
MR	-	-	-	-	-	-	0.0	0.1473	0.0714	0.1190	0.0478	0.3719
NE	-	-	-	-	-	-	-	0.0	0.2446	0.2134	0.1688	0.4894
PA	-	-	-	-	-	-	-	-	0.0	0.1665	0.1293	0.3198
TA	-	-	-	-	-	-	-	-	-	0.0	0.1068	0.3593
TE	-	-	-	-	-	-	-	-	-	-	0.0	0.4474
UR	-	-	-	-	-	-	-	-	-	-	-	0.0

besides using very different writing systems. Standard written Urdu uses Persian, Arabic, and Turkish words heavily, whether adapted phonologically or not.

Given the small amounts of data, it is not surprising that the values of perplexity are different in the two translation directions.

Similarly, standard and written Hindi uses words much more heavily derived or borrowed or even coined from Sanskrit. Despite higher perplexity between these two languages, our approach gives a *+2* increment in the BLEU score, probably because the common core syntax and core vocabulary manifest themselves in every phrase or sentence and thus have higher probabilistic weight. They are, in fact, completely mutually intelligible in the spoken forms and partly in the written form. There are also a lot of Indians who can comfortably read and understand both these languages, even in

their standard, written, and literary forms. The use of WX perhaps allows the models to exploit the core similarities better.

3.5 Ablation Study

This section discusses ablation studies conducted using the proposed method on distant and zero-shot language pairs and back-translation.

3.5.1 Analysis of the proposed approach on distant language pairs

To see whether and to what extent our approach generalizes to more distant language pairs, we also analyze the performance of the proposed approach on (ML↔HI, TA↔HI, and TE↔HI). Malayalam, Tamil, and Telugu belong the Dravidian family, and Hindi is from the Indo-Aryan family. We note that translating between these three Dravidian languages and Hindi still leads to improvement, considering both chrF2 and BLEU scores. The results are shown in Table 3.14.

Table 3.14: Experiments on distant language pairs.

Model	**BLEU**	**chrF2**	**BLEU**	**chrF2**	**BLEU**	**chrF2**
	HI → ML		**HI → TA**		**HI → TE**	
Guzmán et.al [58]	5.12	30	7.57	41	7.19	26
Proposed	3.61	32	**7.86**	44	4.56	27
	ML → HI		**TA → HI**		**TE → HI**	
guzmán et.al [58]	9.08	29	14.55	37	7.97	27
Proposed	**9.96**	33	**15.43**	40	**9.09**	30

3.5.2 Unsupervised settings

We also demonstrate the proposed approach under unsupervised scenarios on zero-shot language pairs, Bhojpuri-Hindi and Magahi-Hindi, for which no parallel training corpora is available. The validation datasets for zero-shot experiments are collected from LoResMT 2020 shared tasks[14]. For training the model, we use NE↔HI language

[14] https://sites.google.com/view/loresmt

pairs and use language transfer on zero-shot pairs to evaluate the model on validation datasets. The reason behind using NE↔HI language pairs for training the model in unsupervised experiments on Bhojpuri-Hindi and Magahi-Hindi is the higher similarity between NE↔HI language pairs with both Bhojpuri-Hindi and Magahi-Hindi zero-shot language pairs based on [100]. The results are shown in Table 3.15, demonstrating the improvement in unsupervised settings also.

Table 3.15: Applying on zero-shot language pairs.

Model	HI → BHO		BHO → HI		HI → MAG		MAG → HI	
	BLEU	**chrF2**	**BLEU**	**chrF2**	**BLEU**	**chrF2**	**BLEU**	**chrF2**
Guzmán et.al [58]	3.34	14	4.58	22	1.67	13	4.86	19
Proposed	3.13	17	**5.72**	27	**2.68**	18	**5.32**	25

3.5.3 Back-translation

Finally we report results on using the approach along with Back-Translation, which has been shown to benefit machine translation for very low resource languages. We selected Gujarati and Hindi language pairs for performing Back-Translation (BT) with the proposed approach. With Back-Translation also, the proposed approach shows an improvement of BLEU point *+0.97* on HI→GU and *+1.36* on GU→HI language pairs, as shown in Table 3.16.

Table 3.16: Experiments on back-translation.

Model	GU→HI				HI→GU			
	BLEU	**chrF2**	**TER**	**WER**	**BLEU**	**chrF2**	**TER**	**WER**
Guzmán et.al [58] + BT(monolingual data)	34.26	55	0.564	58.24	28.32	54	0.619	62.47
Proposed + BT(monolingual data)	35.62	59	0.554	57.39	29.29	58	0.604	61.73

3.6 Summary

In this chapter, we have proposed a simple but effective MT system approach by encoding the source and target script into an intermediate representation, WX-notation,

that helps better modelling. This simple innovation also has the effect of projecting the different languages onto the same orthographic-phonetic character space. This language projection reduces the surface complexity for the algorithm to work on and allows the neural network to model the relationships between languages better to provide an improved translation. Further, we have investigated these results by estimating the similarities and complexities of language pairs and individual languages, respectively, to verify that our results are consistent and agree with the intuitively known facts about the closeness or distances between various language pairs. Moreover, this approach works well under unsupervised settings and works fine for some distant language pairs. The proposed approach leads to improvement over baseline approaches by *0.6* BLEU points to *11.75* BLEU points. The proposed approach has some limitations and boundary conditions. For example, it requires a common transliteration script, which may not available for all morphologically rich languages. Also, we can see from Table 3.14 that this approach does not work well for distant language pairs.