HIT: A Hierarchically Fused Deep Attention Network for Robust Code-mixed Language Representation

Ayan Sengupta[1], **Sourabh Kumar Bhattacharjee**[1],
Tanmoy Chakraborty[2], **Md Shad Akhtar**[2]
[1]Optum Global Advantage (UnitedHealth Group), Noida, India
[2]Dept. of CSE, IIIT-Delhi, India
{ayan.sengupta007, sourabhb398}@gmail.com;
{tanmoy, shad.akhtar}@iiitd.ac.in

Abstract

Understanding linguistics and morphology of resource-scarce code-mixed texts remains a key challenge in text processing. Although word embedding comes in handy to support downstream tasks for low-resource languages, there are plenty of scopes in improving the quality of language representation particularly for code-mixed languages. In this paper, we propose HIT, a robust representation learning method for code-mixed texts. HIT is a hierarchical transformer-based framework that captures the semantic relationship among words and hierarchically learns the sentence-level semantics using a fused attention mechanism. HIT incorporates two attention modules, a multi-headed self-attention and an outer product attention module, and computes their weighted sum to obtain the attention weights. Our evaluation of HIT on one European (*Spanish*) and five Indic (*Hindi*, *Bengali*, *Tamil*, *Telugu*, and *Malayalam*) languages across four NLP tasks on eleven datasets suggests significant performance improvement against various state-of-the-art systems. We further show the adaptability of learned representation across tasks in a transfer learning setup (with and without fine-tuning).

1 Introduction

India is the second most populated country in the world, where ~ 1.36 billion people speak in over 200 different languages. Among them, the top five languages (*Hindi, Bengali, Telegu, Tamil,* and *Malayalam*) covers $\sim 93\%$of the entire population with more than 26% of them being bilingual (as per Wikipedia). Moreover, a significant proportion of them (Singh et al., 2018a) use code-mixed languages to express themselves in Online Social Networks (OSN).

Code-mixing (CM) is a linguistic phenomenon in which two or more languages are alternately used during conversation. One of the languages is usually English, while the other can be any regional language such as Hindi (Hindi + English $\rightarrow$ *Hinglish*), Bengali (Bengali + English $\rightarrow$ *Benglish*), Spanish (Spanish + English $\rightarrow$ *Spaniglish*), etc. Their presence on social media platforms and in day-to-day conversions among the people of a multi-lingual communities (such as Indians) is overwhelming. Despite the fact that a significant population is comfortable with code-mixed languages, the research involving them is fairly young. One of the prime reasons is the linguistic diversity, i.e., research on any language often fails to adapt for other distant languages, thus they need to be studied and researched separately. In recent years, many organizations have identified the challenges and have put in commendable efforts for the development of computational systems in regional monolingual or code-mixed setups.

Traditionally, the NLP community has studied the code-mixing phenomenon from a task-specific point of view. Recently, a few studies (Pratapa et al., 2018; Aguilar and Solorio, 2020) have started learning representations for code-mixed texts for semantic and syntactic tasks. While the former has showcased the importance of multi-lingual embeddings from CM text, the latter has made use of a hierarchical attention mechanism on top of positionally aware character bi-grams and tri-grams to learn robust word representations for CM text. Carrying over the same objective, in this paper, we introduce a novel **HI**erarchically attentive **T**ransformer (HIT) framework to effectively encode the syntactic and semantic features in embeddings space. At first, HIT learns sub-word level representations employing a fused attention mechanism (FAME) – an *outer-product* based attention mechanism (Le et al., 2020) fused with standard multi-headed self-attention (Vaswani et al., 2017). The intuition of sub-word level representation learning is supplemented by the lexical variations of a word in code-mixed languages. The *character-level* HIT helps in representing phonetically similar word and their variations to a similar embedding space and extracts better representation for noisy texts. Subsequently, we apply HIT module at *word-level* to incorporate the semantics at the sentence-level. The output of HIT is a sequence of word representations, and can be fed to the architectures of any downstream NLP tasks. For the evaluation of HIT, we experiment on one classification (sentiment classification), one generative (MT), and two sequence-labelling (POS tagging and NER) tasks. In total, these tasks span to eleven datasets across six code-mixed languages – one European (*Spanish*) and five Indic (*Hindi*, *Bengali*, *Telugu*, *Tamil*, and *Malayalam*). Our empirical results show that representations learned by HIT are superior to existing multilingual and code-mixed representations, and report state-of-the-art performance across all tasks. Additionally, we observe encouraging adaptability of HIT in a transfer learning setup across tasks. The representations learned for a task is employed for learning other tasks w/ and w/o fine-tuning. HIT yields good performance in both setups for two code-mixed languages.

Main contributions: We summarize our contributions as follow:

- We propose a hierarchical attention transformer framework for learning word representations of code-mixed texts for six non-English languages.
- We propose a hybrid self-attention mechanism, FAME, to fuse the multi-headed self-attention and outer-product attention mechanisms in our transformer encoders.
- We show the effectiveness of HIT on eleven datasets across four NLP tasks and six languages.
- We observe good task-invariant performance of HIT in a transfer learning setup for two code-mixed languages.

Reproducibility: Source codes, datasets and other details to reproduce the results have been made public at `https://github.com/LCS2-IIITD/HIT-ACL2021-Codemixed-Representation`.

2 Related Work

Recent years have witnessed a few interesting work in the domain of code-mixed/switched representation learning. Seminal work was driven by bilingual embedding that employs cross-lingual transfer to develop NLP models for resource-scarce languages (Upadhyay et al., 2016; Akhtar et al., 2018; Ruder et al., 2019). Faruqui and Dyer (2014) introduced the BiCCA embedding using bilingual correlation, which performed well on syntactical tasks, but poorly on cross-lingual semantic tasks. Similarly, frameworks proposed by Hermann and Blunsom (2014) and Luong et al. (2015) depend on projecting the words of two languages into a single embedding space.

However, as demonstrated by Pratapa et al. (2018), bilingual embedding techniques are not ideal for CS text processing and should be replaced by multi-lingual embeddings learnt from CM data. The transformer-based Multilingual BERT (Devlin et al., 2019) embedding has been demonstrated (Pires et al., 2019) to possess impressive cross-lingual model transfer capabilities. Also, the XLM model (Conneau and Lample, 2019) has also shown the effects of cross-lingual training for low-resource and CM language tasks.

Another school of thought revolves around sub-word level representations, which can help to capture variations found in CM and transliterated text. Joshi et al. (2016) proposed a CNN-LSTM based model to learn the sub-word embeddings through 1-D convolutions of character inputs. They showed that it resulted in better sentiment classification performance for CM text. On top of this intuition, attention-based frameworks have also been proven to be successful in learning low-level representations. The HAN (Yang et al., 2016) model provides the intuition of hierarchical attention for document classification, which enables it to differentially attend to more and less important content, at the word and sentence levels. In another work, Aguilar and Solorio (2020) proposed CS-ELMo for code-mixed inputs with similar intuition. It utilizes the hierarchical attention mechanism on bi-gram and tri-gram levels to effectively encode the sub-word level representations, while adding positional awareness to it.

Our work builds on top of these two earlier works to push the robustness of code-mixed representations to higher levels. However, the main difference between existing studies and HIT is the incorporation of outer-product attention-based fused attention mechanism (FAME).

3 Proposed Methodology

In this section, we describe the architecture of HIT for learning effective representations in code-

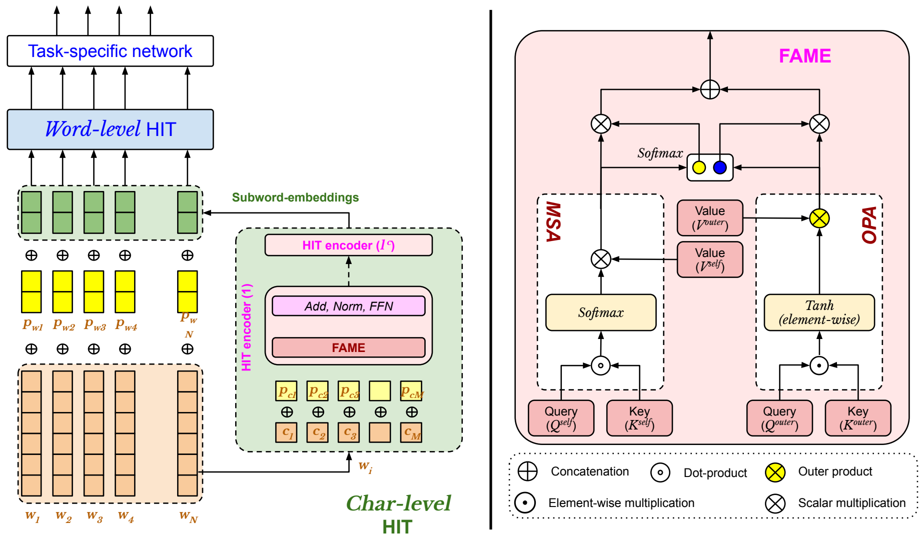


Figure 1: **Hi**erarchical **T**ransformer along with our novel **FAME** mechanism for attention computation.

mixed languages. The backbone of HIT is transformer (Vaswani et al., 2017) and Hierarchical Attention Network (HAN) (Yang et al., 2016). HIT takes a sequence of words (a code-mixed sentence) $S = \langle w_1, w_2, ..., w_N \rangle$ as input and processes each word w_i using a *character-level* HIT to obtain sub-word representation $S^{sb} = \langle sb_1, sb_2, ..., sb_N \rangle$. The *character-level* HIT is a transformer encoder, where instead of computing multi-headed self-attention only, we amalgamate it with an outer-product attention mechanism (Le et al., 2020) as well. The intuition of outer-product attention is to extract higher-order character-level relational similarities among inputs. To leverage both attention mechanisms, we compute their weighted sum using a softmax layer. Subsequently, we pass it through the typical *normalization* and *feed-forward* layers to obtain the encoder's output. A stacking of l_c encoders is used. In the next layer of the hierarchy, these sub-word representations are combined with positional and rudimentary embeddings of each word and forwarded to the *word-level* HIT's encoder. Finally, the output of *word-level* HIT is fed to the respective task-specific network. The hierarchical nature of HIT enables us to capture both *character-level* and *word-level* relational (syntactic and semantic) similarities. A high-level schema of HIT is shown in Figure 1.

3.1 Fused Attention Mechanism (FAME)

FAME extends the multi-headed self-attention (MSA) module of a standard transformer by including a novel outer-product attention (OPA) mechanism. Given an input x, we use three weight matrices, W_Q^{self}, W_K^{self}, and W_V^{self}, to project the input to *Query* (Q^{self}) , *Key* (K^{self}), and *Value* (V^{self}) representations for MSA, respectively. Similarly for OPA we use W_Q^{outer}, W_K^{outer}, and W_V^{outer} for the projecting x to Q^{outer}, K^{outer} and V^{outer}. Next, the two attention mechanisms are learnt in parallel, and a weighted sum is computed as its output. Formally, $H = \alpha_1 \cdot Z_{self} \oplus \alpha_2 \cdot Z_{outer}$, where Z_{self} and Z_{outer} respectively are the outputs of multi-headed self attention and outer-product attention modules, and α_1 and α_2 are the respective weights computed through a softmax function.

Multi-Headed Self Attention. The standard transformer self-attention module (Vaswani et al., 2017) computes a scaled dot-product between the *query* and *key* vectors prior to learn the attention weights for the *value* vector. We compute the output as follows:

$$\begin{aligned} Z_{self} &= softmax\left(\frac{Q^{self} \cdot K^{self^T}}{\sqrt{d^k}}\right) V^{self} \\ &= \sum_i^N softmax\left(\frac{q \cdot k_i}{\sqrt{d^k}}\right) v_i, \forall q \in Q^{self} \end{aligned}$$

where N is the sequence length, and d^k is the dimension of the *key* vector.

Outer-Product Attention. This is the second attention that we incorporate into FAME. Although the fundamental process of OPA (Le et al., 2020) is similar to the multi-headed self-attention computation, OPA differs in terms of different operators and the use of row-wise *tanh* activation instead of softmax. To compute the interaction between the query and key vectors, we use element-wise multiplication as opposed to the dot-product in MSA. Subsequently, an element-wise *tanh* is applied before computing the outer-product with the value vector. The intuition is to exploit fine-level associations between the *key*-scaled *query* and *value* representations in a code-mixed setup. Similar to the earlier case, we define OPA as:

$$Z_{outer} = \sum_{i}^{N} tanh\left(\frac{q \odot k_i}{\sqrt{d_k}}\right) \otimes v_i, \forall q \in Q^{outer}$$

where $\odot$ is the element-wise multiplication, and $\otimes$ is the outer-product.

3.2 Task-specific Layers

As we mention earlier, HIT can be adapted for various NLP tasks including sequence labelling, classification, or generative problems. In the current work, we evaluate HIT on part-of-speech (POS) tagging, named-entity recognition (NER), sentiment classification, and machine translation (MT). We mention their specific architectural details below.

For the sentiment classification, we apply a *GlobalAveragePooling* operation over the token embeddings to obtain the sentence embeddings. Additionally, we concatenate extracted statistical features along with the embeddings before feeding into the final classification layer. We use *tf-idf* (term frequency–inverse document frequency) vectors for $\{1, 2, 3\}$-grams of words and characters extracted from each text. We hypothesize that these statistical features contain sufficient information to get rid of any handcrafted features like the ones suggested by Bansal et al. (2020). Finally, a *softmax* activation function is used for the prediction. Similarly, for POS tagging and NER, the corresponding labels for each of the token's embedding is obtained through a softmax activated output. In case of MT, we use an encoder-decoder framework where both the encoder and the decoder are based on the HIT framework.

4 Experiments, Results, and Analyses

In this section, we furnish the details of chosen datasets, our experimental results, comparative study, and necessary analyses.

4.1 Datasets

We evaluate 11 publicly available datasets across 4 tasks in 6 code-mixed languages. For POS tagging, we employ Hindi, Telugu, Bengali, and Spanish, whereas, we evaluate Hindi and Spanish datasets for NER. Similarly, in sentiment classification, we incorporate Hindi, Tamil, Malayalam, and Spanish code-mixed sentences. Finally, for machine translation, we use a recently released Hindi-English code-mixed parallel corpus. A brief statistics of all datasets is presented in Table 1.

Tasks	Lang	Train		Test		Total	#Labels
		#Sent	#Token	#Sent	#Token		
POS	Hi*	1191	6575	148	2300	1489	14
	Te*	1,585	7,190	198	2,927	1,982	52
	Be*	500	4,108	62	631	626	39
	Sp	27,893	11,897	4,298	3,866	36,489	17
NER	Hi*	1663	9,397	207	3,272	2,079	7
	Sp	33,611	52,680	10,085	23,787	53,781	19
Sentiment	Hi*	3,103	9,005	387	3,191	3,879	3
	Ta	11,335	27,476	3,149	10,339	15,744	4
	Ma	4,851	16,551	1,348	6,028	6,739	4
	Sp	12,194	28,274	1,859	7,822	15,912	3
MT	En (Src)	248,330	84,609	2,000	5,314	252,330	-
	Hi (Tgt)		108,442		5,797		

Table 1: Dataset statistics. Star(*) signifies 90-10 ratio.

• **POS tagging:** We use the Hindi-English code-mixed POS dataset provided by Singh et al. (2018b). It was collected from Twitter and has 1489 sentences. Each token in the sentence is tagged with one of the 14 tags[1]. The Bengali and Telugu datasets are collected from ICON-2016 workshop[2]. The instances are the social-media messages, collected from Twitter, Facebook and WhatsApp, and have 1982 and 626 sentences in Telugu and Bengali, respectively. These two datasets follow Google universal tagset (Petrov et al., 2011) and contain 52 and 39 tags respectively. For Spanish, we use Linguistic Code-switching Evaluation (LinCE) POS dataset (AlGhamdi et al., 2016) consisting of more that $35k$ sentences with 14 tags.

• **Sentiment classification:** We explore the Hinglish sentiment classification dataset developed by Joshi et al. (2016). The dataset contains 3879 Facebook public posts comprises of 15% *negative*, 50% *neutral*, and 35% *positive* samples. We further consider two sentiment classification datasets for Dravidian languages *viz.* Tamil and Malayalam (Chakravarthi et al., 2020), containing 15744 and 6739 instances respectively with four sentiment labels – *positive*, *negative*, *neutral*, and *mixed feelings*. Additionally, we use SemEval-2020 (Patwa et al., 2020) dataset for Spanish code-mixed sentiment classification. It supports a classic 3-way sentiment classification.

• **Named-entity recognition:** For NER, we em-

[1]We furnish the details of tagset in the appendix

[2]`http://amitavadas.com/Code-Mixing.html`

Model	Hindi			Tamil			Malayalam			Spanish		
	Pr.	Re.	F1	Pr.	Re.	F1	Pr.	Re.	F1	Pr.	Re.	F1
BiLSTM	0.916	0.901	0.909	0.502	0.428	0.451	0.653	0.588	0.612	0.429	0.431	0.428
Subword-LSTM	0.905	0.907	0.905	0.503	0.418	0.426	0.577	0.592	0.581	0.445	0.437	0.432
HAN	0.915	0.906	0.908	0.490	0.411	0.439	0.639	0.611	0.634	0.449	0.439	0.440
ML-BERT	0.919	**0.914**	0.909	0.260	0.310	0.280	0.600	0.630	0.610	0.451	0.419	0.437
CS-ELMO	0.921	0.903	0.909	0.515	0.432	0.459	0.666	0.623	0.642	0.429	0.453	0.431
HIT	**0.956**	0.914	**0.915**	0.499	**0.451**	**0.473**	0.710	0.628	0.651	**0.502**	**0.454**	**0.460**
(-) *Atn*^{outer}	0.933	0.911	0.913	**0.520**	0.448	0.455	**0.718**	0.624	**0.655**	0.463	0.440	0.445
(-) *char-level* HIT	0.903	0.887	0.901	0.504	0.418	0.432	0.659	0.605	0.627	0.448	0.438	0.433

Table 2: Performance of HIT on sentiment classification. Best scores are highlighted in bold.

ploy Hindi (Singh et al., 2018c) and Spanish (Aguilar et al., 2018) datasets with 2079 and 52781 sentences, respectively. In Hindi, the labels are *name*, *location*, and *organization*. The Spanish dataset has six additional labels – *event*, *group*, *product*, *time*, *title*, and *other* named entities.

• **Machine Translation:** We utilize a recently developed Hindi-English code-mixed parallel corpus for machine translation (Gupta et al., 2020) comprising more than $200k$ sentence pair. For experiments, we transliterate all Devanagari Hindi text.

4.2 Baselines

POS tagging, NER & sentiment classification: ▷ **BiLSTM** (Hochreiter and Schmidhuber, 1997): It is a weak baseline with two conventional BiLSTM layers. For POS and NER, we additionally incorporate a CRF layer for the final classification. ▷ **HAN** (Yang et al., 2016): We adapt the Hierarchical Attention Network (HAN) for our purpose. The subword embedding is computed at the first level of attention network followed by a word-level attention at the second level. Recently, Bansal et al. (2020) also adopted HAN for code-mixed classification. ▷ **ML-BERT** (Devlin et al., 2019): We fine-tune multilingual BERT (Devlin, 2019). ▷ **CS-ELMo** (Aguilar and Solorio, 2020): It is one of state-of-the-arts on code-mixed languages. It uses pre-trained ELMo (Peters et al., 2018) to transfer knowledge from English to code-mixed languages. ▷ **Subword-LSTM** (Joshi et al., 2016): It is a hybrid CNN-LSTM model. A 1D convolution operation is applied for the subword representation. Subsquently, the convoluted features are max-pooled and fed to an LSTM. Since this system disregards word boundaries in a sentence, we use it for *sentiment classification only*.

Machine translation: For machine translation, we evaluate HIT against **GFF-Pointer** (Gupta et al., 2020), a gated feature fusion (GFF) based approach to amalgamate the XLM and syntactic features during encoding and a Pointer generator for decoding. Furthermore, we also incorporate three other baselines for comparison – **Seq2Seq** (Sutskever et al., 2014), **Attentive-Seq2Seq** (Bahdanau et al., 2014) and **Pointer Generator** (See et al., 2017).

4.3 Experimental Setup

For each experiment, we use a $dropout = 0.1$ in both transformer block and the task specific layers. Categorical cross-entropy loss with Adam ($\eta = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$) optimizer (Kingma and Ba, 2014) is employed in all experiments. We train our models for maximum 500 epochs with an early-stopping criteria having $patience = 50$. We additionally use a learning rate scheduler to reduce learning rate to 70% at plateaus with a patience of 20 epochs. All models are trained with *batch-size*= 32.

4.4 Experimental Results

We compute precision, recall, F1-score for POS, NER, and sentiment classification, whereas, BLEU, METEOR, and ROUGE scores are reported for the machine translation task.

Sentiment classification: As shown in Table 2, HIT obtains best F1-scores across all languages. For Hindi, three baselines (BiLSTM, ML-BERT, and CS-ELMo) obtain the best F1-score of 0.909, where HIT yields a small improvement with 0.915 F1-score. In comparison, we observe an improvement of 1.4% for Tamil, where HIT and the best baseline (CS-ELMo) report 0.473 and 0.459 F1-scores, respectively. We observe the same pattern for Malayalam and Spanish as well – in both cases, HIT obtains improvements of 0.9% and 2.0%, respectively. For Malayalam, HIT reports 0.651 F1-score, whereas CS-ELMo reports 0.642 F1-score. In case of Spanish, HAN turns out to be the best baseline with 0.440 F1-score. Com-

Model	Hindi			Telugu			Bengali			Spanish		
	Pr.	Re.	F1	Pr.	Re.	F1	Pr.	Re.	F1	Pr.	Re.	F1
BLSTM-CRF	0.821	0.913	0.782	0.595	0.747	0.572	0.842	0.851	0.817	0.704	0.836	0.680
HAN	0.802	0.879	0.815	0.693	0.701	0.684	0.811	0.823	0.818	0.497	0.629	0.527
ML-BERT	0.833	0.884	0.847	0.802	0.762	0.771	0.793	0.815	0.807	0.853	0.808	0.802
CS-ELMO	0.885	**0.961**	0.910	0.831	0.790	0.775	**0.873**	0.851	0.847	0.740	**0.835**	0.729
HIT	**0.918**	0.955	**0.919**	0.815	0.749	0.762	0.841	**0.855**	**0.853**	**0.871**	0.822	**0.825**
(-) Atn^{outer}	0.893	0.948	0.914	**0.839**	**0.793**	**0.786**	0.839	0.852	0.845	0.859	0.813	0.820
(-) *char-level* HIT	0.686	0.922	0.708	0.629	0.758	0.626	0.802	0.830	0.819	0.723	0.796	0.732

Table 3: Performance of HIT on POS tagging. Best scores are highlighted in bold.

Model	Hindi			Spanish		
	Pr.	Re.	F1	Pr.	Re.	F1
BLSTM-CRF	0.622	0.781	0.579	0.581	0.659	0.603
HAN	0.721	0.767	0.695	0.615	0.679	0.644
ML-BERT	0.792	0.779	0.714	0.652	0.623	0.643
CS-ELMO	0.815	0.780	0.735	0.683	0.668	0.671
HIT	**0.829**	0.788	**0.745**	**0.695**	**0.671**	**0.684**
(-) Atn^{outer}	0.821	0.767	0.732	0.669	0.663	0.668
(-) *char-level* HIT	0.556	**0.815**	0.528	0.498	0.664	0.539

Table 4: Performance of HIT on NER.

Model	B	R	M
Seq2Seq†	15.49	35.29	23.72
Attentive-Seq2Seq†	16.55	36.25	24.97
Pointer Generator†	17.62	37.32	25.61
GFF-Pointer†	21.55	40.21	28.37
Transformer	21.83	42.19	27.89
HIT	**28.22**	**51.52**	**29.59**
(-) Atn^{outer}	25.95	49.19	27.63

Table 5: Performance of HIT encoder-decoder on En-Hi Translation (**B**: BLEU, **R**: Rouge-L and **M**: METEOR). † Values are taken from Gupta et al. (2020).

paratively, HIT achieves 0.460 F1-score. The last two rows of Table 2 report ablation results – a) excluding outer-product attention (Atn^{outer}) from HIT; and b) excluding sub-word embeddings (*character-level* HIT). In all cases, the absence of *sub-word* embeddings yields negative effect on the performance; hence, suggesting the effectiveness of *character-level* HIT in the architecture. On the other hand, omitting outer-product attention declines F1-scores in 3 out of 4 cases – we observe a margin improvement of 0.04 points for Malayalam. In summary, HIT attains state-of-the-art performance across all four datasets, whereas, the best baseline (CS-ELMo) reports 1.2% lower scores on average.

POS tagging: Table 3 shows the comparative results for POS tagging in Hindi, Telugu, Bengali, and Spanish. Similar to sentiment classification, we observe that HIT attains best F1-scores across three datasets (0.625% better on average). It achieves 0.919, 0.762, 0.853, and 0.825 F1-scores for Hindi, Telugu, Bengali, and Spanish, respectively. In comparison, CS-ELMo yields best F1-scores among all the baselines across three datasets *viz.* Hindi (0.910), Telugu (0.775), and Bengali (0.847). For Spanish, ML-BERT obtains the best baseline F1-score of 0.802. From ablation, we observe the negative effect on performance by removing either the outer-product attention or *character-level* HIT for the majority of the cases.

NER: The performance of HIT for NER is also in-line with the previous two tasks, as show in Table 4. As mentioned earlier, we evaluate HIT for Hindi and Spanish datasets. In both cases, we observe $\geq 1\%$ improvement in F1-score, in comparison with the best baseline (CS-ELMo).

In all three tasks, CS-ELMo is arguably the most consistent baseline. Together with the state-of-the-art performance of HIT, we regard the good performance to the subword-level contextual modelling – both systems use contextual representational models (ELMo and Transformer) to encode the syntactic and semantic features. Moreover, the FAME module in HIT assists in improving the performance even further.

Machine Translation: Finally, Table 5 reports the results for the English to Hindi (En-Hi) machine translation task. For comparison, we also report BLEU, METEOR, and ROUGE-L scores for four baseline systems – Seq2Seq (Sutskever et al., 2014), Attentive-Se2Seq (Bahdanau et al., 2014), Pointer Generator (See et al., 2017), and GFF-Pointer (Gupta et al., 2020). For all three metrics, HIT reports significant improvement (1-9 points) over the state-of-the-art and other baselines. GFF-Pointer obtains 21.55 BLEU score, while the other baselines yield BLEU scores in the range $[15 - 17]$. In comparison, HIT obtain 28.22 BLEU, *an extremely convincing result.* Similarly, HIT reports 51.52 ROUGE and 29.59 METEOR scores, respectively.

		Fine-tune	Target Tasks		
			PoS	NER	Sentiment
Source Tasks	PoS	w/o	0.919	0.702	**0.890**
		w/		0.578	0.863
	NER	w/o	**0.924**	0.745	**0.885**
		w/	0.873		**0.893**
	Sentiment	w/o	**0.936**	0.729	0.871
		w/	**0.928**	0.691	

(a) Hindi code-mixed

		Fine-tune	Target Tasks		
			PoS	NER	Sentiment
Source Tasks	PoS	w/o	0.825	**0.710**	0.417
		w/		0.656	0.419
	NER	w/o	**0.881**	0.648	**0.473**
		w/	0.663		**0.446**
	Sentiment	w/o	**0.918**	**0.969**	0.445
		w/	0.732	0.687	

(b) Spanish code-mixed

Table 6: Transfer learning models. Code-mixed word representations, learned for a (source) task, is utilized for building models for other (target) tasks of same language w/ and w/o fine-tuning. We highlight the cases in bold where transfer learning achieves better performance than original base HIT.

	Pos	Neg	Neu
Pos	**0.85**	0.05	**0.10**
Neg	0.03	**0.87**	**0.10**
Neu	-	0.02	**0.98**

(a) Sentiment

	B-Per	I-Per	B-Loc	I-Loc	B-Org	I-Org	O
B-Per	**0.85**	0.01	-	-	-	-	**0.14**
I-Per	0.03	**0.87**	-	-	-	-	**0.10**
B-Loc	-	-	**0.91**		0.02	0.02	0.05
I-Loc	-	0.08		**0.85**	-	-	0.07
B-Org	0.02	0.02	-	-	**0.82**	-	**0.14**
I-Org	-	-	-	-	-	**0.71**	**0.29**
O	-	-	-	-	**0.47**	-	**0.53**

(b) NER

Table 7: Confusion matrices (in %) for sentiment and NER on Hindi code-mixed dataset[3].

	Input	Gold	Prediction	
			HIT	CS-ELMo
1	**Org:** *safal videsh yatra ke liye badhai ho sir* **Trans:** *Congratulations on the successful foreign trip sir*	Pos	Pos	Neu
2	**Org:** *desh chodo pahaley yeh media ko change karo ... !! ?* **Trans:** *Leave the country, first change the media*	Neu	Neg	Neg

(a) Sentiment

Input	**Org:** *@gurmeetramrahim {dhan dhan satguru}$_{Per}$ tera hi aasra #msgloveshumanity salute 2 {msg}$_{Org}$ <url>* **Translated:** *@gurmeetramrahim we depend on you {dhan dhan satguru}$_{Per}$ #msgloveshumanity salute 2 {msg}$_{Org}$ <url>*
Prediction	**HIT:** *@gurmeetramrahim {dhan dhan satguru}$_{Per}$ tera hi aasra #msgloveshumanity salute 2 msg <url>* **CS-ELMo:** *@gurmeetramrahim {dhan dhan satguru}$_{Per}$ tera hi aasra #msgloveshumanity salute 2 {msg}$_{Org}$ <url>*

(b) NER

Table 8: Error Analysis on Hindi code-mixed dataset.

4.5 Effects of Transfer Learning across Tasks

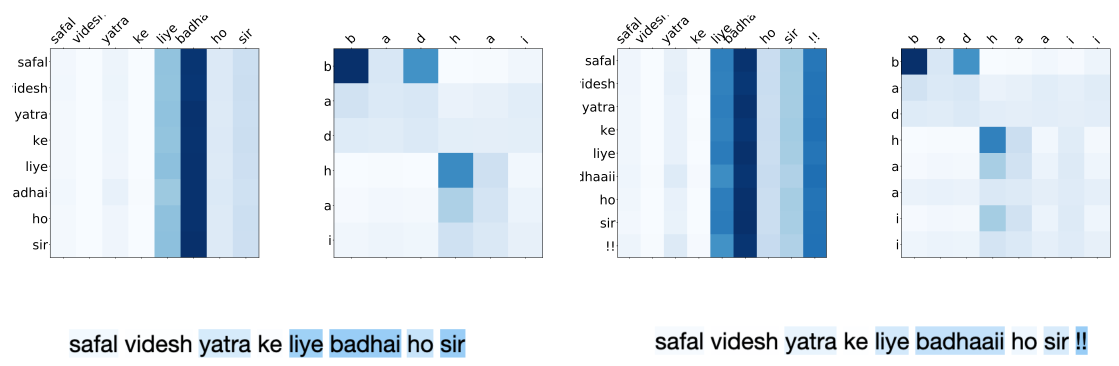
One of the core objectives of representation learning is that the learned representation should be task-invariant – the representations learned for one task should also be (near) effective for other tasks. The intuition is that the syntactic and semantic features captured for a language should be independent of the tasks, and if it does not comply, the representation can be attributed to capture the task-specific feature, instead of linguistic features. To this end, we perform transfer learning experiments with (w/) and without (w/o) fine-tuning. Since we have only one dataset for Tamil, Telugu, Bengali, and Malayalam, we choose Hindi and Spanish code-mixed datasets (POS, NER, and sentiment classification) for the study. Table 6 reports results for both code-mixed languages. For each case, we learn HIT's representation on one (source) task and subsequently utilize the representation for the other two tasks (targets). Moreover, we repeat each experiment with and without fine-tuning HIT.

For Hindi code-mixed, we do not observe the positive effect of transfer learning for NER. It could be because of the limited lexical variations of named-entities in other datasets. However, we obtain the best F1-score (0.936) for POS tagging in a transfer learning setup with sentiment classification. Similarly, for the sentiment classification as target, we observe performance improvements with both POS and NER as source tasks. In Spanish, we also observe increment in F1-scores for all three tasks. We attribute these improvements to the availability of more number of sentences for HIT to leverage the linguistic features in both Hindi and Spanish.

4.6 Error Analysis

In this section, we analyze the performance of HIT both quantitatively and qualitatively. At first, we report the confusion matrices[3] for Hindi NER and sentiment classification in Table 7. In both cases, we observe the *true-positives* to be significant for all labels. Furthermore, we also observe the *false-positives* to be extremely low (except for '*B-Org*'

[3]Confusion matrices and more error cases for other tasks are presented in the appendix.



(a) Original input.

(b) Perturbed input.

Figure 2: Interpretation of Hindi code-mixed sentiment prediction (a) Grad-CAM (Selvaraju et al., 2019) analysis of original text; (b) Grad-CAM of perturbed text (*'badhaaii'*); For both case, the *word-level* and *char-level* attention plots are shown.The impact of the word *'badhai'* on the overall prediction is shown through gradients and attention heatmaps. The representation learning of phonetically similar word (e.g., *'badhaaii'*) is also noticeable in the perturbed case. It signifies that HIT is flexible to the spelling variations, a common feature in code-mixed environment.

in NER) for majority of the cases – suggesting very good precision in general. The major contribution in error is due to the *neutral* and *other* classes in sentiment and NER, respectively. In sentiment analysis, 10% of the *positive* and *negative* labels each were mis-classified as *neutral*. Similarly in NER, we observe that the *organization* entities (*B-Org & I-Org*) and *other* classes confuse with each other in a significant number of samples. It may be due to the under-represented ($\sim 13\%$) *organization* entities in the dataset.

We also perform qualitative error analysis of HIT and CS-ELMo. Table 8 reports the results for the NER and sentiment classification tasks[3]. For the first example in sentiment classification, HIT accurately predicts the sentiment labels as *positive*; however in comparison, CS-ELMo mis-classifies it as *neutral*. For the second example, both HIT and CS-ELMo wrongly predict the sentiment as *neutral*. One possible reason could be the presence of the negatively-inclined word *chodo* (*leave*) in the sentence. For NER, the sentence has two entities (one *person* and one *organization*). While HIT correctly identifies '*dhan dhan satguru*' as person, it could not recognize '*msg*' as organization. On the other hand, CS-ELMo correctly identifies both.

Furthermore, we take the first example of sentiment analysis (from Table 8) to get the insight of HIT. It is not hard to understand that the most positive vibe is attributed by the phrase '*badhai ho sir*' (*congratulations sir*). To validate our hypothesis, we use a gradient based interpretation technique, Grad-CAM (Selvaraju et al., 2019), which uses gradients of neural networks to show the effect of neurons on the final output. Due to hierarchical and modular nature of HIT, we are able to extract the intermediate word level representations learnt by the *character-level* HIT and compute the gradient of loss of the actual class considering these representations. The magnitude of gradient shows the impact of each word on the final output. Figure 2a shows the word-level and character-level gradient maps for the original input. We can observe that HIT attends to the most important component in both cases. For *word-level*, it highlights the positive phrase '*liye badhai*' (*congratulations on*). Moreover, character-level HIT attends to the two syllables '*b*' and '*dh*' in the word '*badhai*' (*congratulation*). It suggests that both the *word-level* and *character-level* components are capable of extracting important features from inputs. Furthermore, to check the robustness, we investigate HIT on a perturbed input. In the previous example, we tweak the spelling of the most important word '*badhai*' to '*badha$\underline{a}\underline{i}$i*' (an out-of-vocabulary word considering the dataset). Figure 2b shows similar patterns in the perturbed input case as well. It signifies that HIT identifies the phonetic similarity of the two words and is flexible to the spelling variants, a common feature in code-mixed environment.

4.7 HIT's Performance on Monolingual Data

In this section, we outline the performance of HIT for monolingual and low-resource settings. We consider the sentiment classification dataset curated by Akhtar et al. (2016), containing 5417 transliterated

Model	Hi Sentiment			Magahi POS		
	Pr.	Re.	F1	Pr.	Re.	F1
BiLSTM	0.619	0.533	0.554	0.594	0.804	0.626
HAN	0.602	0.528	0.551	0.729	0.857	0.649
ML-BERT	0.604	0.556	0.576	0.757	0.867	0.708
CS-ELMO	0.593	0.520	0.542	0.771	0.884	0.759
HIT	**0.641**	**0.629**	**0.635**	**0.783**	**0.913**	**0.775**

Table 9: Performance of HIT on monolingual tasks. Best scores are highlighted in bold.

Hindi reviews with 4 sentiment labels - *positive*, *negative*, *neutral*, and *conflict*. We also utilize a Magahi POS dataset (Kumar et al., 2012), annotated with 33 tags from the BIS-tagset [4]. We report the performance of HIT and other baselines on these two datasets in Table 9. For the Hindi sentiment classification task, we observe that HIT yields an F1-score of 0.635, which is better than CS-ELMo and ML-BERT by 9.3% and 5.9%. Also, for Magahi POS, HIT reports the best F1-score of 0.775 – increaments of $+2.1\%$ and $+9.5\%$ over CS-ELMo and ML-BERT, respectively. These results suggest that HIT is capable of handling monolingual and low-resource texts in an efficient manner.

4.8 Learnable Parameters and Power Usage

We conduct all our experiments on 1 Tesla T4 GPU. In Table 10, we report the total trainable parameters for HIT and other baselines. We observe that HIT requires a comparable number of parameters. For instance, in the Hindi-English sentiment analysis task (sequence classification), HIT has a total ~$2.7M$ trainable parameters, while other baselines such as, CS-ELMo, HAN, Subword-LSTM, and BiLSTM require ~$2.9M$, ~$2.7M$, ~$2.1M$, and ~$2.8M$ parameters, respectively. ML-BERT has a whopping ~$179.2M$ parameters. Similarly, in Hindi-English POS tagging, the number of parameters for HIT is comparable (or even lesser) – HIT: ~$1.4M$, CS-ELMo: ~$2.4M$, HAN: ~$1.4M$, BiLSTM-CRF: ~$1.5M$, ML-BERT: ~$177.9M$. We observed similar distribution for other tasks/languages as well.

We further note that HIT is significantly more efficient than the current SOTA models as it takes 13 s/epoch to train which is significantly lower than CS-ELMo (18 s/epoch), HAN (14 s/epoch), and ML-BERT (172 s/epoch), while it takes a bit more time compared to BiLSTM (12 s/epoch) and Subword-LSTM (7 s/epoch). We also computed the amount of power consumption for training HIT for a maximum 500 epochs. Following the guidelines of Strubell et al. (2019), we estimate a total power consumption of 0.383 kWh and equivalent CO2 emission of 0.365 pounds.

[4] https://thottingal.in/blog/2019/09/10/bis-pos-tagset-review/

Tasks	Model	# Params(M)	Train Time(s/ep)
Sentiment	CS-ELMo	2.9	18
	HAN	2.7	14
	BiLSTM	2.8	12
	ML-BERT	179.2	172
	HIT	2.7	13
POS	CS-ELMo	2.4	4
	HAN	1.4	3
	BiLSTM-CRF	1.5	2
	ML-BERT	177.9	114
	HIT	1.4	2

Table 10: Parameters and Runtime: Number of trainable parameters and training runtime (second/epoch) for the Hi-En PoS and sentiment classification tasks.

5 Conclusion

In this work, we present HIT – a hierarchical transformer-based framework for learning robust code-mixed representations. HIT contains a novel fused attention mechanism, which calculates a weighted sum of the multi-headed self attention and outer-product attention, and is capable of capturing relevant information at a more granular level. We experimented with eleven code-mixed datasets for POS, NER, sentiment classification, and MT tasks across six languages. We observed that HIT successfully outperforms existing SOTA systems. We also demonstrate the task-invariant nature of the representations learned by HIT via a transfer learning setup, signifying it's effectiveness in learning linguistic features of CM text rather than task-specific features. Finally, we qualitatively show that HIT successfully embeds semantically and phonetically similar words of a code-mixed language.

Acknowledgement

The work was partially supported by the Ramanujan Fellowship (SERB) and the Infosys Centre for AI, IIITD.

References

Gustavo Aguilar, Fahad AlGhamdi, Victor Soto, Mona Diab, Julia Hirschberg, and Thamar Solorio. 2018. Named entity recognition on code-switched data: Overview of the CALCS 2018 shared task. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 138–147, Melbourne, Australia. Association for Computational Linguistics.

Gustavo Aguilar and Thamar Solorio. 2020. From English to code-switching: Transfer learning with strong morphological clues. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8033–8044, Online. Association for Computational Linguistics.

Md Shad Akhtar, Asif Ekbal, and Pushpak Bhattacharyya. 2016. Aspect based sentiment analysis in Hindi: Resource creation and evaluation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2703–2709, Portorož, Slovenia. European Language Resources Association (ELRA).

Md Shad Akhtar, Palaash Sawant, Sukanta Sen, Asif Ekbal, and Pushpak Bhattacharyya. 2018. Solving data sparsity for aspect based sentiment analysis using cross-linguality and multi-linguality. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 572–582, New Orleans, Louisiana. Association for Computational Linguistics.

Fahad AlGhamdi, Giovanni Molina, Mona Diab, Thamar Solorio, Abdelati Hawwari, Victor Soto, and Julia Hirschberg. 2016. Part of speech tagging for code switched data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 98–107, Austin, Texas. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Srijan Bansal, Vishal Garimella, Ayush Suhane, Jasabanta Patro, and Animesh Mukherjee. 2020. Code-switching patterns can be an effective route to improve performance of downstream NLP applications: A case study of humour, sarcasm and hate speech detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1018–1023, Online. Association for Computational Linguistics.

Bharathi Raja Chakravarthi, Vigneshwaran Muralidaran, Ruba Priyadharshini, and John Philip McCrae. 2020. Corpus creation for sentiment analysis in code-mixed Tamil-English text. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 202–210, Marseille, France. European Language Resources association.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32, pages 7059–7069. Curran Associates, Inc.

Jacob Devlin. 2019. Multilingual bert.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471, Gothenburg, Sweden. Association for Computational Linguistics.

Deepak Gupta, Asif Ekbal, and Pushpak Bhattacharyya. 2020. A semi-supervised approach to generate the code-mixed text using pre-trained encoder and transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2267–2280, Online. Association for Computational Linguistics.

Karl Moritz Hermann and Phil Blunsom. 2014. Multilingual models for compositional distributed semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 58–68, Baltimore, Maryland. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Aditya Joshi, Ameya Prabhu, Manish Shrivastava, and Vasudeva Varma. 2016. Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2482–2491.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Ritesh Kumar, Bornini Lahiri, and Deepak Alok. 2012. Developing a POS tagger for Magahi: A comparative study. In *Proceedings of the 10th Workshop on Asian Language Resources*, pages 105–114, Mumbai, India. The COLING 2012 Organizing Committee.

Hung Le, Truyen Tran, and Svetha Venkatesh. 2020. Self-attentive associative memory. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5682–5691. PMLR.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159, Denver, Colorado. Association for Computational Linguistics.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Parth Patwa, Gustavo Aguilar, Sudipta Kar, Suraj Pandey, Srinivas PYKL, Dan Garrette, Bj"orn Gamb"ack, Tanmoy Chakraborty, Thamar Solorio, and Amitava Das. 2020. SemEval-2020 Sentimix Task 9: Overview of SENTIment Analysis of Code-MIXed Tweets. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Slav Petrov, Dipanjan Das, and Ryan T. McDonald. 2011. A universal part-of-speech tagset. *CoRR*, abs/1104.2086.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Adithya Pratapa, Monojit Choudhury, and Sunayana Sitaram. 2018. Word embeddings for code-mixed language processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3067–3072, Brussels, Belgium. Association for Computational Linguistics.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *J. Artif. Int. Res.*, 65(1):569–630.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2019. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359.

Kushagra Singh, Indira Sen, and Ponnurangam Kumaraguru. 2018a. Language identification and named entity recognition in Hinglish code mixed tweets. In *Proceedings of ACL 2018, Student Research Workshop*, pages 52–58, Melbourne, Australia. Association for Computational Linguistics.

Kushagra Singh, Indira Sen, and Ponnurangam Kumaraguru. 2018b. A Twitter corpus for Hindi-English code mixed POS tagging. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 12–17, Melbourne, Australia. Association for Computational Linguistics.

Vinay Singh, Deepanshu Vijay, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018c. Named entity recognition for hindi-english code-mixed social media text. In *Proceedings of the seventh named entities workshop*, pages 27–35.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Advances in neural information processing systems. In *Neural Information Processing Systems (Nips)*, pages 3104–3112.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. 2016. Cross-lingual models of word embeddings: An empirical comparison. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1670, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

A Appendix

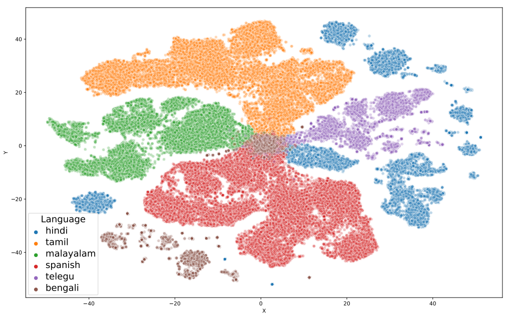
A.1 Semantic Understanding of Languages

In this section, we study the semantic relationships between different Indic languages. We calculate the proportion of common words in Table 11 between different language pairs to understand the multilingualism in India. We observe that Bengali code-mixed texts have the highest proportion of English words 32% as compared to other languages. Moreover, 50% of all Bengali words are also present in the Hindi CM texts, although 58% of those words are English. We observe that users using Hindi CM texts use very few words taken from other languages. On the other hand, a significant proportion of Bengali and Telugu CM words are common in other languages, although, majority of them are English. The two Dravidian languages, Tamil and Malayalam, show a very distinctive behavior. They share very little linguistic similarity with other Indic languages. On the other hand, 10% of all Tamil words are used in Malayalam and 17% of all Malayalam words are used in Tamil. Moreover, this sharing is not driven by English, as, only 27% of these words are English, which is the lowest proportion among all other language pairs. Being originate from a similar root and having a phonetic resemblance makes Tamil and Malayalam *sister languages*[5]. Similar observations are also made from the word representation lens. We use t-SNE (Van der Maaten and Hinton, 2008) plots to embed HIT's representations onto a 2-D space for interpretability (Fig 3). Although, the embeddings are well clustered based on the languages, we can easily figure out the semantically similar words across languages embedded onto a similar space. Furthermore, Fig 3(b) shows that pronouns (e.g., '*aap*') in Tamil, Telegu and Hindi are embedded onto a similar space with Bengali words *'aamar'*, '*aamay*'. Although each of these representations are learned on separate models on separate datasets, the robustness of the underlying hierarchical representation enables our model to capture cross-lingual semantics from noisy code-mixed texts. We can attribute these observations to the relatedness of Indic languages on a socio-cultural basis.

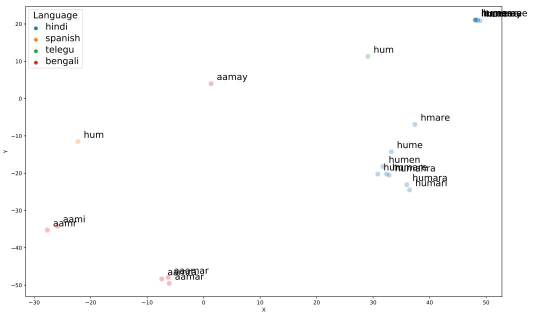
A.2 Datasets

We report all available POS tags in Table 12.

[5]https://royalsocietypublishing.org/doi/10.1098/rsos.171504



(a)



(b)

Figure 3: t-SNE visualization (a) of all words; (b) of selected pronouns. Overlapping clusters show how semantically similar words from different languages are embedded onto a similar space.

A.3 Confusion Matrices and Error Analysis

We report the confusion matrices to show the label-wise performance for the sentiment classification, PoS tagging and NER in Tables 5, 4, and 6, respectively.

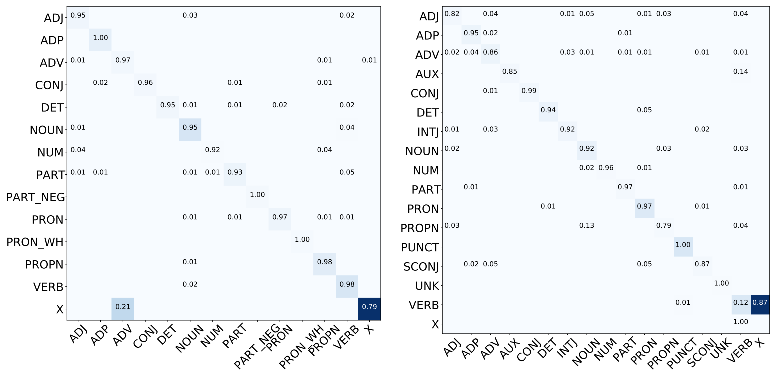
We similarly perform qualitative analysis on the MT task where our model shows superior performance as compared to the baselines. In example 1 of Table 13 (d), HIT translates the English text '*'Licencing and import policies were liberalise"* to *"Licencing aur policies liberal the* । *"*. Although this prediction has very low BLEU score when evaluated against the target, this example shows an interesting observation. The overall translation is a contextually meaningful sentence in Hindi. Further HIT translates the phrase *'were liberalise'* to *'liberal the'*. In Hindi, *'the'* represents past tense. Another interesting observation is the ability of HIT to copy texts from source to predicted text. Even without having an explicit copying mechanism (See et al., 2017), HIT is able to understand

	Target Language					
Source Language	**Hindi (English)**	**Malayalam (English)**	**Tamil (English)**	**Bengali (English)**	**Telegu (English)**	**Spanish (English)**
Hindi	1.00 (0.16)	0.02 (0.41)	0.04 (0.39)	0.02 (0.58)	0.02 (0.57)	0.07 (0.62)
Malayalam	0.14 (0.41)	1.00 (0.06)	0.17 (0.27)	0.03 (0.71)	0.05 (0.57)	0.07 (0.64)
Tamil	0.15 (0.39)	0.10 (0.27)	1.00 (0.07)	0.03 (0.69)	0.05 (0.56)	0.07 (0.64)
Bengali	0.50 (0.58)	0.16 (0.58)	0.23 (0.69)	1.00 (0.32)	0.21 (0.71)	0.36 (0.72)
Telegu	0.36 (0.57)	0.15 (0.57)	0.29 (0.56)	0.12 (0.71)	1.00 (0.22)	0.28 (0.65)
Spanish	0.12 (0.62)	0.02 (0.64)	0.03 (0.64)	0.02 (0.72)	0.03 (0.65)	1.00 (0.11)

Table 11: Proportion of words in source language in the target language.

Lang	**POS tags**
Hindi (14)	*X, VERB, NOUN, ADP, PROPN, ADJ, PART, PRON, DET, ADV, CONJ, PART_NEG, PRON_WH, NUM*
Bengali (39)	*N_NN, V_VM, RD_PUNC, N_NNP, PSP, PR_PRP, JJ, RB_AMN, CC, QT_QTF, DM_DMD, RP_RPD, @, RD_RDF, V_VAUX, DT, PR_PRQ, #, RP_NEG, E, $, RB_ALC, N_NNV, PR_PRL, N_NST, RP_INJ, RD_SYM, DM_DMR, RP_INTF, PR_PRF, DM_DMQ, QT_QTO, U, QT_QTC, PR_PRC, RD_ECH, QY_QTO, Ã°Å¸Ëœ, ∼*
Telugu (52)	*N_NN, N_NNP, RD_RDF, RD_PUNC, V_VM, JJ, @, PSP, PR_PRP, RP_INJ, DT, RB_AMN, CC, $, U, E, #, N_NNV, &, PR_PRQ, V_VAUX, RD_PUNC", ∼, RD_RDFP, QT_QTF, RD_UNK, DM_DMD, RP_RPD, RB_ALC, DM_DMQ, RD_ECH, N_NST, acro, PR_PRL, QT_QFC, RP_RDF, PR_PRC, r, RD_SYM, RD_RDFF, psp, PR_PRF, QT_QTP, RD_P/UNC, PR_PPR, PR_RPQ, RPR_PRP, RP_INTF, -*
Spanish (17)	*VERB, PUNCT, PRON, NOUN, DET, ADV, ADP, INTJ, CONJ, ADJ, AUX, SCONJ, PART, PROPN, NUM, UNK, X*

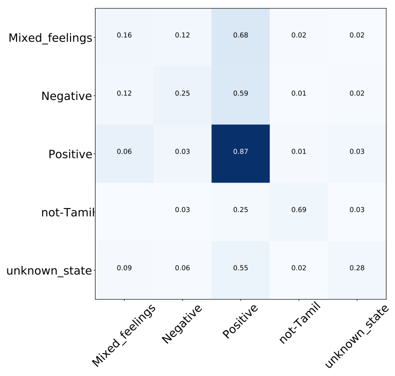
Table 12: POS tagsets for different datasets.



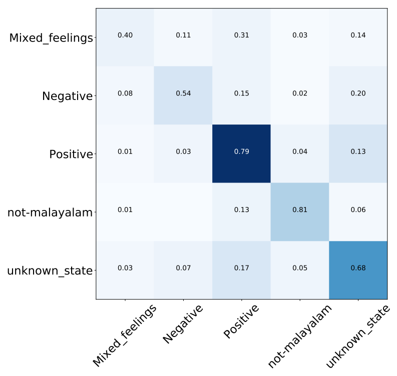
(a) Hindi (b) Spanish

Figure 4: Confusion matrices of `HIT` on POS tasks. Due to high cardinality of output classes, we do not report for Bengali and Telugu.

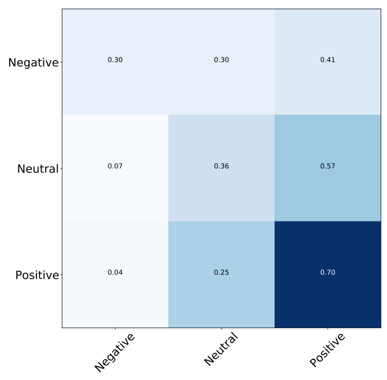
the key phrases that co-occur in both Hindi and English, like, numeric and proper nouns and copies these tokens while generating. This shows how our model can also be used in conditional generation of texts. It also ends the sentence with ।, which is a common punctuation widely used as a full stop in Hindi texts.



(a) Tamil

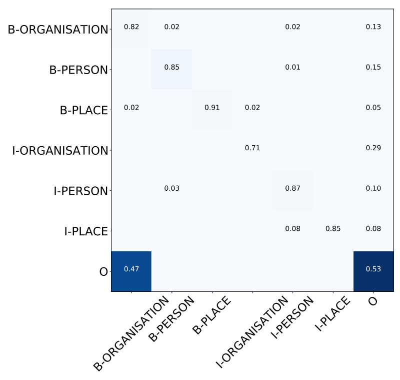


(b) Malayalam

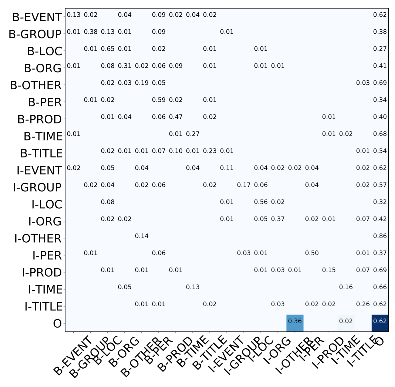


(c) Spanish

Figure 5: Confusion matrices of `HIT` on sentiment tasks.



(a) Hindi



(b) Spanish

Figure 6: Confusion matrices of `HIT` on NER.

	Input	Sys	**Prediction**
1	**Org:** #surgicalstrike_X #pakistan_X will_V not_Neg sleep_N in_ADP peace_N tonight_N ._X khamoshi_V toofan_V ke_ADP aane_V ki_ADP aahat_N to_P nahi_Neg	A	#surgicalstrike_X #pakistan_X will_V not_part_Neg sleep_N in_ADP peace_N tonight_N ._X, khamoshi_V toofan_V ke_ADP aane_V ki_ADP aahat_N to_P nahi_part_neg
	Translated: #surgicalstrike #pakistan will not sleep in peace tonight. Does this silence signify that a storm is approaching	B	#surgicalstrike_X #pakistan_X will_V not_part_Neg sleep_N in_ADP peace_N tonight_N ._X, khamoshi_V toofan_V ke_N aane_V ki_ADP aahat_ADP to_P nahi_part_Neg
2	**Org:** minimum_N cincuenta_Num mil_Num por_ADP persona_N ._Punct	A	minimum_N cincuenta_Num mil_Num por_ADP persona_N ._Punct
	Translated: minimum fifty thousand per person .	B	minimum_N cincuenta_Num mil_Num por_ADP persona_N ._Punct

(a) PoS

	Input	Sys	**Prediction**
1	**Org:** @gurmeetramrahim {dhan dhan satguru}$_{Per}$ tera hi aasra #msgloveshumanity salute 2 {msg}$_{Org}$ <url>	A	@gurmeetramrahim {dhan dhan satguru}$_{Per}$ tera hi aasra #msgloveshumanity salute 2 msg <url>
	Translated:	B	@gurmeetramrahim {dhan dhan satguru}$_{Per}$ tera hi aasra #msgloveshumanity salute 2 {msg}$_{Org}$ <url>
2	**Org:** ste {sábado}$_{Time}$ nuestras alumnas en {imagen modeling}$_{Org}$ by {la gatita}$_{Per}$ reciben la visita de {monic abbad}$_{Per}$, joven . . . <url>	A	ste {sábado}$_{Time}$ nuestras alumnas en imagen modeling by la gatita reciben la visita de {monic}$_{Per}$ abbad , joven . . . <url>
	Translated: This saturday our students in image modeling by the kitten receive a visit from young monic abbad	B	ste {sábado}$_{Time}$ nuestras alumnas en imagen modeling by la gatita reciben la visita de {monic abbad}$_{Per}$, joven . . . <url>

(b) NER

	Input	**Gold**	**Prediction**	
			A	**B**
1	**Org:** *safal videsh yatra ke liye badhai ho sir* **Trans:** *Congratulations on the successful foreign trip sir*	Pos	Pos	Neu
2	**Org:** *nunca pensé que " bruh " me frustraría tanto* **Trans:** *I never thought that "bruh" would frustrate me so much*	Neu	Neu	Neg
3	**Org:** *desh chodo pahaley yeh media ko change karo ... !! ?* **Trans:** *Leave the country, first change the media*	Neu	Neg	Neg

(c) Sentiment

1	**Source:** *Licencing and import policies were liberalise* **Reference:** *license tatha import ki policies ko udar banaya gaya*
	HIT: Licencing aur policies liberal the ।
2	**Source:** *This fact is based on possibility* **Reference** *yah fact possibility par aadharit hai ।*
	HIT: yah fact possibility par aadharit hai

(d) MT

Table 13: Error Analysis. System A denotes HIT and B denotes CS-ELMO.