ARJUN: A Parameter-Efficient Multi-Objective Pre-Training Framework for Language Models

Anonymous ACL submission

Abstract

The proliferation of large-scale language models has significantly advanced natural language processing capabilities, yet their substantial computational requirements create barriers for deployment in resource-constrained environments and limit accessibility for underrepresented language communities. We present a novel multi-objective pre-training approach for developing parameter-efficient language models. This addresses the critical challenge of computational efficiency for low-resource languages while maintaining competitive performance. Our method, implemented in our ARJUN model family, strategically combines four complementary pre-training objectives within a unified transformer encoder-decoder framework: (1) Masked Token Prediction (MT) following BERT's bidirectional understanding paradigm, (2) Next Token Prediction (NT) incorporating autoregressive generation capabilities, (3) Masked Span Prediction (MS), and (4) Complete Sentence Prediction (CS) adapted from T5 that bridge discriminative and generative tasks. Using this multi-objective approach, Arjun achieves competitive or superior performance for downstream applications with only 44M parameters, compared to models that are 2.5-12.5 times larger in diverse tasks in three Indian languages – Bangla, Tamil, and Hindi. Comprehensive evaluation on 12 downstream tasks demonstrates that Arjun variants establish new state-of-the-art performance on multiple benchmarks, with particularly an average Macro-F1 percentage point improvement of 1.75 in Bangla, 3.10 in Tamil and 2.12 in Hindi. Ablation studies confirm that joint training with all four objectives consistently outperforms single-objective and partial combinations, with improvements ranging from 0.5-2.0 percentage points across tasks, thereby validating the synergistic effects of our multi-objective framework.

1 Introduction

The rapid advancement of large pre-trained language models has revolutionized natural language processing, with models like BERT, GPT, and T5 achieving remarkable performance across diverse tasks (Devlin et al., 2019; Raffel et al., 2020). However, these models require billions of parameters, creating substantial barriers for deployment in resource-constrained environments. Recent research has explored parameter-efficient approaches to address these limitations, including adapter-based methods, prefix tuning, and low-rank adaptation (LoRA (Hu et al., 2021)). While these techniques reduce the computational overhead of fine-tuning large models, they still rely on massive pre-trained backbones and do not fundamentally address the efficiency challenges at the pre-training stage. Simultaneously, multi-objective pre-training has emerged as a promising paradigm for developing more versatile language representations (Ke et al., 2024; Doddapaneni et al., 2022). Studies have shown that combining complementary training objectives can lead to models that better capture both local and global linguistic patterns, resulting in improved generalization across downstream tasks.

Building on these insights, we propose a novel multi-objective pre-training framework that achieves strong performance with significantly fewer parameters. Our approach centers on the joint optimization of four carefully designed pre-training objectives, each targeting different aspects of language understanding. Figure 1 illustrates our overall architecture, showing how these four objectives—Masked Token Prediction (MT), Next Token Prediction (NT), Masked Span Prediction (MS), and Complete Sentence Prediction (CS)—interact within a unified transformer encoder-decoder framework. The MT objective, inspired by BERT's masked language modeling, en-

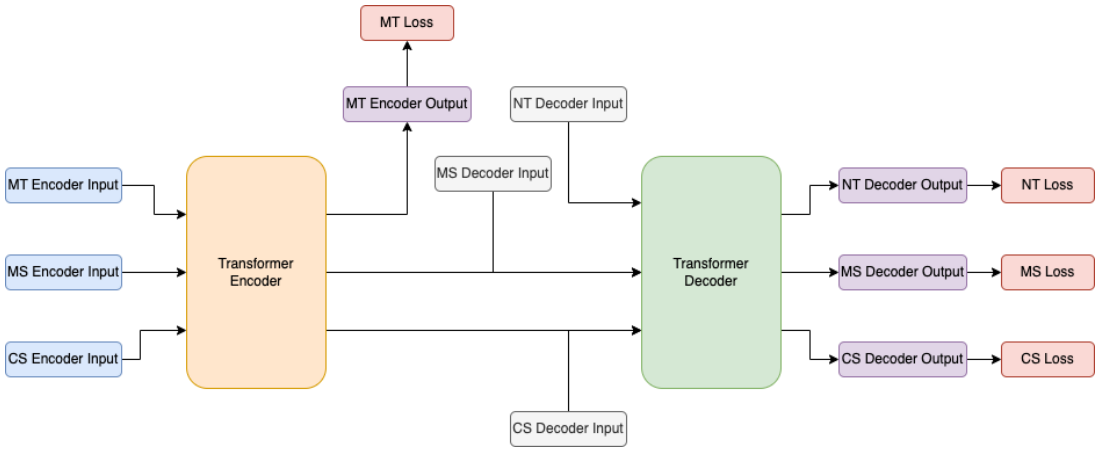

Figure 1: Overall architecture of the Arjun model showing how the four pre-training objectives (MT, NT, MS, and CS) interact with the transformer encoder-decoder components. The Arjun is trained jointly optimized for the MT, MS, CS and NT loss. Equation 5 is the mathematical expression of the joint loss. Table 1 depicts the data to be processed for the multi-objective model training.

ables bidirectional context understanding (Devlin et al., 2019). The NT objective, following GPT's autoregressive approach (Radford et al., 2018), captures causal dependencies and generation capabilities. The MS and CS objectives, adapted from T5's (Raffel et al., 2020) span corruption strategy, teaches the model to handle variable-length semantic units that bridge discriminative and generative tasks. MT provides bidirectional contextual understanding, NT teaches sequential generation patterns, MS captures multi-token semantic dependencies, and CS develops sentence-level coherence.

We implement our framework in **Arjun**, a family of transformer encoder-decoder models with 69M total parameters. Despite this relatively modest size, Arjun demonstrates remarkable effectiveness across multiple Indian languages. We train separate models for different corpora: Vac-Arjun on the Vācaspati Bangla corpus (Bhattacharyya et al., 2023), and language-specific models (Indic-BangArjun, Indic-TamArjun, Indic-HindArjun) on respective IndicCorp v2 datasets (Kakwani et al., 2020). This approach allows us to capture language-specific nuances while maintaining computational efficiency.

Our empirical evaluation spans 12 diverse downstream tasks across Bangla, Tamil, and Hindi, including sentiment analysis, named entity recognition, topic classification, and specialized tasks such as poem classification and spell error detection. The results demonstrate that our parameter-efficient Arjun models consistently achieve competitive or superior performance compared to much (5-10x) larger models. Notably, they outperform multilingual models, including XLM-R base (280M) (Conneau et al., 2020), IndicBERT-v2 variants (278M) (Doddapaneni et al., 2022), and language-specific models like BanglaBERT (110M) (Bhattacharjee et al., 2022). The contributions of this work are three-fold:

1. We introduce a novel multi-objective pre-training framework that strategically combines four complementary objectives to create parameter-efficient yet highly effective language models. The code is available at https://anonymous.4open.science/r/arjun-multi-task-learning-E290.
2. We demonstrate through a comprehensive evaluation that joint optimization of these objectives leads to superior performance compared to single-objective approaches, with our 44M parameter models achieving results competitive with models up to 10-20x larger for downstream supervised tasks.
3. We provide thorough analysis through ablation studies showing that each component contributes meaningfully to the overall performance, validating the design choices of our multi-objective approach.

2 Related Work

The landscape of pre-trained language models has been shaped by several foundational approaches and subsequent innovations. BERT introduced masked language modeling as a powerful pre-training objective, achieving bidirectional context understanding through its encoder-only architec-

ture (Devlin et al., 2019). RoBERTa improved upon BERT by removing the next sentence prediction objective, using dynamic masking, and training with larger batch sizes and more data, demonstrating that careful optimization of training procedures can significantly enhance performance (Liu et al., 2019). The T5 framework revolutionized the field by casting all NLP tasks into a unified text-to-text format, employing span corruption as its primary pre-training objective within an encoder-decoder architecture (Raffel et al., 2020). Building on autoregressive foundations, XLNet proposed permutation language modeling to combine the benefits of autoregressive and autoencoding approaches, achieving bidirectional context modeling without relying on masked tokens (Yang et al., 2019). Parameter efficiency became a critical concern with ALBERT, which introduced factorized embedding parameterization and cross-layer parameter sharing to reduce BERT's parameter count by up to 89% while maintaining competitive performance (Lan et al., 2020). More recent work has explored sophisticated multi-objective frameworks: ELECTRA introduced replaced token detection as an alternative to masked language modeling, training models to distinguish real tokens from generated replacements (Clark et al., 2020), while UL2 proposed a unified framework combining multiple denoising objectives through mixture-of-denoisers, demonstrating universal effectiveness across diverse NLP tasks and achieving state-of-the-art results on over 50 benchmarks (Tay et al., 2022). GLM presented autoregressive blank infilling with 2D positional encodings, successfully unifying natural language understanding and generation capabilities within a single pre-training framework (Du et al., 2022). For Indian languages specifically, the AI4Bharat initiative has contributed significantly with IndicBERT models trained on comprehensive Indian language corpora, while MuRIL has targeted multilingual Indian language understanding (Kakwani et al., 2020; Khanuja et al., 2021). Our work builds upon these advances by proposing a novel multi-objective pre-training approach that strategically combines complementary objectives to achieve strong performance with exceptional parameter efficiency.

3 Methodology

Our goal is to pre-train a language model with parameters θ that captures rich linguistic and semantic representations by jointly optimizing multiple training objectives. Let $x = [x_1, x_2, \ldots, x_T]$ be an input sequence sampled from the corpus. Figure 1 illustrates the overall architecture of our proposed Arjun model. We define the following 4 pre-training objectives.

3.1 Masked Token Prediction (MT)

Following BERT, we randomly select a subset of token positions $M \subset \{1, 2, \ldots, T\}$ for masking, typically 15% of the tokens. We replace the tokens at these positions with a special [MASK] token to create a corrupted sequence $\tilde{x}$. The objective is to minimize:

$$L_{\text{MT}}(\theta, x) = -\sum_{i \in M} \log p_\theta(x_i | \tilde{x}) \quad (1)$$

Unlike BERT, we do not implement the Next Sentence Prediction (NSP) objective, as prior experiments (Liu et al., 2019) have demonstrated its insignificant contribution to model performance.

3.2 Next Token Prediction (NT)

Following the GPT approach, this is an autoregressive language modeling objective where the model predicts each token given its preceding context:

$$L_{\text{NT}}(\theta, x) = -\sum_{t=1}^{T} \log p_\theta(x_t | x_1, x_2, \ldots, x_{t-1}) \quad (2)$$

3.3 Masked Span Prediction (MS)

Instead of masking individual tokens, we randomly select spans of consecutive tokens for masking. We mask 15% of the words (not tokens) in the input, with consecutive masks appended to create single spans. Each span is replaced with a unique sentinel token (e.g., [S1], [S2]) to help the model predict spans in the correct order. The objective is:

$$L_{\text{MS}}(\theta, x) = -\sum_{s \in S} \log p_\theta(x_s | \tilde{x}) \quad (3)$$

where x_s denotes the original sequence of tokens in span s.

3.4 Complete Sentence Prediction (CS)

We pick a random token position in a sentence. On the encoder side, tokens from the beginning of the sentence up to the chosen position are provided as input. On the decoder side, the model is trained to generate the rest of the sentence:

$$L_{\text{CS}}(\theta, x) = -\sum_{t=k+1}^{T} \log p_\theta(x_t | x_1, x_2, \ldots, x_k) \quad (4)$$

where k is the randomly chosen token position in the sentence.

3.5 Joint Training and Model Architecture

The multi-objective pre-training framework Arjun employs a transformer encoder-decoder architecture that processes different types of inputs depending on the training task, with MT utilizing only the encoder, NT using only the decoder, and MS and CS leveraging both components to develop comprehensive language representations. All the four tasks are jointly trained within each mini-batch of our training process. For every set of sentences, we simultaneously create four different sets of inputs and outputs corresponding to these objectives as shown in Table 1. Note that while English is used here for clarity of demonstration, our actual implementation uses the respective scripts of Bangla, Tamil, and Hindi languages.

Eq. (5) represents the joint loss to train Arjun, where λ_{MT}, λ_{NT}, λ_{MS}, and λ_{CS} are hyperparameters that control the contribution of each objective to the final loss.

$$\begin{aligned} L_{\text{joint}}(\theta, x) = \lambda_{\text{MT}} \cdot L_{\text{MT}}(\theta, x) + \lambda_{\text{NT}} \cdot L_{\text{NT}}(\theta, x) \\ + \lambda_{\text{MS}} \cdot L_{\text{MS}}(\theta, x) + \lambda_{\text{CS}} \cdot L_{\text{CS}}(\theta, x) \end{aligned} \quad (5)$$

Our goal is to find the optimal model parameters θ^* that minimize the expected joint loss over the corpus:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{x \sim D}[L_{\text{joint}}(\theta, x)] \quad (6)$$

This integrated approach ensures the model learns to develop robust representations that capture both bidirectional context (primarily from encoder-focused tasks) and generative capabilities (from decoder-focused tasks), while efficiently sharing parameters across all training signals.

4 Experiments and Results

4.1 Data

For our pre-training experiments, we utilized multiple corpora as shown in Table 2. We name the model pretrained on the Vācaspati (Bhattacharyya et al., 2023) dataset as Vac-Arjun. Additionally, we have trained monolingual models on the Bangla, Tamil, and Hindi corpora from the IndicCorp v2 Dataset (Kakwani et al., 2020). We name these models as Indic-Bangla-Arjun, Indic-Tamil-Arjun, and Indic-Hindi-Arjun respectively. We have used the same architecture and hyperparameters for pre-training on all these corpora.

4.2 Supervised Tasks

Table 3 summarizes the 12 datasets used for evaluating our models across three languages. The Bangla datasets offer comprehensive evaluation across literary, social media, and general text domains, while Tamil datasets focus on news and entity recognition. Hindi evaluation is the most diverse with 5 distinct tasks spanning news classification, product and movie reviews, discourse analysis, and NER. The dataset sizes vary considerably, from smaller specialized collections (Poem Classification with 1,491 samples) to larger NER datasets (WikiANN Tamil with 25,663 samples). Most datasets are derived from real-world sources such as social media, news articles, and literary works, ensuring our evaluation reflects diversity and practical language understanding challenges.

4.3 Pre-training Implementation Details

Our multi-objective pre-trained model, Arjun, employs a Transformer encoder-decoder architecture (Vaswani et al., 2017) with 6 layers in both the encoder and decoder. We set the hidden layer dimension to 512, the feed-forward network dimension to 2048, and utilized 8 attention heads. A dropout rate of 0.1 was applied throughout the model to prevent overfitting. We have used AdamW optimizer with an initial learning rate of 5e-5 with a linear decay. For input processing, we limited the maximum sequence length to 256 tokens. In our models, we share the same weight matrix between the two embedding layers and the pre-softmax linear transformation, similar to (Press and Wolf, 2016). For each pre-trained model (Vac-Arjun, Indic-BangArjun, Indic-TamArjun, and Indic-HindArjun), we trained a separate WordPiece tokenizer (Devlin et al., 2019) on the respective corpora, each with a vocabulary size of 50,000 tokens. The models were trained with a batch size of 256 for 20 epochs, resulting in approximately 900K training steps for the Vācaspati-based model, with proportionally adjusted steps for the IndicCorp-based models according to their corpus sizes. The total model size is 69M parameters for each model, encompassing

Task	Encoder Input	Encoder Output	Decoder Input	Decoder Output
MT	[MT] Arjun model [MASK] optimized for [MASK] tasks	[-100] [-100] [-100] is [-100] [-100] multiple [-100]	-	-
MS	[MS] Arjun model [SPAN1] for multiple [SPAN2]	-	[SOS] [SPAN1] is optimized [SPAN2] tasks	[SPAN1] is optimized [SPAN2] tasks [EOS]
NT	-	-	[NT] Arjun model is optimized for multiple tasks	Arjun model is optimized for multiple tasks [EOS]
CS	[CS] Arjun model	-	[SOS] is optimized for multiple tasks	is optimized for multiple tasks [EOS]

Table 1: Examples of encoder-decoder inputs and outputs for our four pre-training tasks using a sample sentence "Arjun model is optimized for multiple tasks". For MT, encoder outputs show token indices where [-100] indicates masked positions not used in loss computation.

Corpus	Sentences	Words
Vācaspati (Bangla)	11M	115M
IndicCorp v2 - Bangla	19M	898M
IndicCorp v2 - Tamil	12M	483M
IndicCorp v2 - Hindi	105M	6,113M

Table 2: Statistics of pre-training corpora used in our experiments.

both the encoder and decoder components. For efficient training, we utilized an AWS p4de.24xlarge instance (featuring 8 NVIDIA A100 GPUs with 40GB memory each and 1,152GB system RAM) working in parallel, completing the pre-training process in approximately 7-20 days.

4.4 Fine-tuning Setup

For downstream task evaluation, we focused exclusively on the discriminative tasks. During fine-tuning, we discarded the decoder and attached a task-specific classification layer (softmax or sigmoid, depending on the task requirements) to the encoder outputs. This approach resulted in models with 44M parameters for all fine-tuning experiments. For all experiments, we employed the Macro-F1 score with a threshold of 0.5 as our primary evaluation metric. We used AdamW optimizer with a constant learning rate of 5e-6 to fine-tune all the models.

Multilingual Models. We benchmarked our approach against several established multilingual models: XLM-R base (280M parameters) (Conneau et al., 2020), XLM-R large (550M parameters), mBERT (180M parameters) (Pires et al., 2019), IndicBERT-v1 (18M parameters) (Kakwani et al., 2020), various IndicBERT-v2 (Doddapaneni et al., 2022) variants (278M parameters), and MuRIL (236M parameters) (Khanuja et al., 2021). These multilingual models were evaluated across all three languages: Bangla, Tamil, and Hindi.

Bangla Task Evaluation. For Bangla language tasks, we compared our models (Vac-Arjun and Indic-BangArjun, both 44M parameters) against language-specific models including BanglaBERT (110M parameters) (Bhattacharjee et al., 2022), BanglaBERT Large (336M parameters), sahajBERT (18M parameters) (Diskin et al., 2021), and VAC-BERT (17M parameters) (Bhattacharyya et al., 2023).

Tamil and Hindi Task Evaluation. For Tamil language tasks, we evaluated our Indic-TamArjun model (44M parameters) against the multilingual models mentioned above, as well as Tamil-specific models including TamilBERT and Tamil-SQuAD-BERT by L3Cube Pune (Joshi, 2022), both with 278M parameters. Similarly, for Hindi language tasks, we compared our Indic-HindArjun model (44M parameters) against the multilingual models as well as HindiBERT and Hindi-SQuAD-BERT by L3Cube Pune (Joshi, 2022), also with 278M parameters.

4.5 Results

We evaluate our multi-objective pre-training approach across three Indic languages: Bangla, Tamil, and Hindi. For each language, we compare our models against state-of-the-art multilingual and language-specific models on various downstream tasks. All results are reported as the average of 10 independent runs, with statistical significance calculated using a threshold of $p < 0.005$.

4.5.1 Bangla Supervised Tasks

Table 4 presents a comprehensive comparison of our proposed models, Vac-Arjun and Indic-BangArjun, against existing state-of-the-art models across five downstream tasks for Bengali language processing. The results demonstrate that our multi-objective pre-training approach achieves competitive or superior performance across all benchmark

Language	Dataset Name	Dataset Size			ML Task	Source	Description
		Train	Valid	Test			
Bangla	Poem Classification	1,192	-	299	Sentence-level 5-class classification	Vācaspati (Bhattacharyya et al., 2023)	Classification of Rabindranath Tagore's poems into five categories: pūjā (devotion), prema (love), prākṛiti (nature), svadesha (nationalism), and vichitrā (miscellaneous).
	Sentiment 2-class	8,612	-	2,154	Sentence-level binary classification	(Sazzed, 2020)	Reviews annotated on YouTube Bangla drama with positive and negative labels.
	Sentiment 3-class	12,145	-	1,581	Sentence-level 3-class classification	(Islam et al., 2021)	Social media comments with positive, negative, and neutral labels from 13 domains.
	WikiANN Bangla	20,223	2,985	2,690	Named entity recognition	(Pan et al., 2017)	Bangla subset of WikiANN with 4 entity classes: person, location, organization, and other.
	Spelling Error Detection	15,237	-	3,810	Sentence-level binary classification	Vācaspati (Bhattacharyya et al., 2023)	Detection of context-sensitive spelling errors.
Tamil	News Category Classification	5,346	669	669	Sentence-level 3-class classification	IndicNLPSuite (Kakwani et al., 2020)	Prediction of genre/topic of news articles (entertainment, sports, business, etc.).
	WikiANN Tamil	20,466	2,586	2,611	Named entity recognition	(Pan et al., 2017)	Tamil subset of WikiANN with 4 entity classes: person, location, organization, and other.
Hindi	BBC News Classification	3,467	-	866	Sentence-level 14-class classification	IndicNLPSuite (Kakwani et al., 2020)	Genre/topic classification for news articles into 14 categories.
	WikiANN Hindi	9,463	1,114	1,256	Named entity recognition	(Pan et al., 2017)	Hindi subset of WikiANN with 4 entity classes: person, location, organization, and other.
	IIT Patna Movie Review	2,480	310	310	Sentence-level 3-class classification	IndicNLPSuite (Kakwani et al., 2020)	Movie review sentiment analysis with positive, negative and neutral classes. Combined from IIT-Patna, IIT-Bombay and iNLTK datasets.
	IIT Patna Product Review	4,182	523	523	Sentence-level 3-class classification	IndicNLPSuite (Kakwani et al., 2020)	Product reviews with positive, negative and neutral sentiment labels.
	MIDAS Discourse	7,974	997	997	Sentence-level 6-class classification	(Dhanwal et al., 2020)	Classification into 6 discourse categories: argumentative, descriptive, dialogic, informative, narrative and other.

Table 3: Overview of datasets used for evaluating our models across Bangla, Tamil, and Hindi languages

tasks, with our models collectively establishing new state-of-the-art performance on four out of five tasks.

Indic-BangArjun achieves the best results in three-class sentiment classification (71.30%), poem classification (67.51%), and NER (94.63%), while Vac-Arjun demonstrates superior performance in spell error detection (82.45%). For binary sentiment classification, IndicBERT-v2-MLM-Sam-TLM slightly outperforms our models with a score of 95.75%, though both of our models remain highly competitive in this task (95.40% and 95.17% respectively). The strong performance of Indic-BangArjun on poem classification is particularly noteworthy, showing a substantial improvement over previous state-of-the-art models.

These results are especially impressive considering that both Vac-Arjun and Indic-BangArjun have only 44M parameters each, substantially fewer than many competing models such as BanglaBERT (110M), BanglaBERT Large (336M), and the IndicBERT-v2 variants (278M). Our models outperform or remain competitive with much larger models that have 5-10 times more parameters, demonstrating the efficiency and effectiveness of our multi-objective pre-training approach.

4.5.2 Tamil and Hindi Supervised Tasks

The promising results observed for Bangla extend to Tamil and Hindi as well. As shown in Table 5 and Table 6, our models consistently achieve competitive or superior performance across multiple tasks in these languages while maintaining the same parameter-efficient architecture.

For Tamil, our Indic-TamArjun model achieves state-of-the-art results on both News Category Classification (98.15%) and NER (90.08%), outperforming specialized Tamil language models like TamilBERT (97.89%, 83.35%) and Tamil-SQuAD-BERT (97.42%, 83.41%), as well as larger multilingual models.

For Hindi, Indic-HindArjun demonstrates exceptional performance on BBC News Classification (62.06%, a 6.27 percentage point improvement over the next best model), NER (91.39%), and MIDAS Discourse (69.41%). For the IIT Patna sentiment analysis tasks, our model achieves competitive performance (71.16% and 79.06%).

Similar to our observations with Bangla models, both Indic-TamArjun and Indic-HindArjun achieve these results with only 44M parameters, approximately 5-6 times smaller than most competing models (236-278M parameters). Some larger models, notably XLM-V (780M parameters) and certain configurations of IndicBERT-v2, significantly un-

Model	Parameters	Sentiment Analysis		Poem (5-class)	NER (4-class)	Spell Error
		2-class	3-class			
mBERT	180M	94.80	63.68	59.85	86.51	41.00
MuRIL	236M	95.00	68.73	52.36	90.62	42.50
XLM-R (base)	270M	94.64	67.80	35.00	87.50	42.50
XLM-R (large)	550M	93.89	65.75	31.86	87.60	40.00
XLM-V	780M	95.23	70.04	25.20	89.59	43.46
IndicBERT-v1	18M	91.86	59.34	43.67	88.31	41.50
IndicBERT-v2-SS	278M	91.34	60.50	12.88	82.88	66.35
IndicBERT-v2-MLM-only	278M	95.24	70.74	65.37	93.10	79.70
IndicBERT-v2-MLM-Back-TLM	278M	85.30	70.61	65.20	92.73	80.25
IndicBERT-v2-MLM-Sam-TLM	278M	**95.75**	70.42	62.39	92.24	80.67
BanglaBERT	110M	95.00	68.68	54.40	91.52	78.90
BanglaBERT Large	336M	95.04	71.11	63.20	90.46	82.08
sahajBERT	18M	93.75	67.52	46.34	85.50	70.89
VAC-BERT	17M	94.75	68.79	60.39	89.70	77.72
Indic-BangArjun **(Ours)**	**44M**	95.40	**71.30**	**67.51**	**94.63**	81.83
Vac-Arjun **(Ours)**	**44M**	95.17	70.06	65.23	93.57	**82.45**

Table 4: Performance comparison (Macro-F1 scores) of our proposed models with state-of-the-art models across five Bengali language tasks. Our multi-objective pre-training approaches achieve competitive or superior results across all tasks despite having significantly fewer parameters than many competing models. All results are reported as averages from 10 independent runs. **Bold** indicates results that are better with statistical significance ($p < 0.005$).

Model	Parameters	Tasks	
		News Category Classification	NER
mBERT	180M	93.27	78.20
MuRIL Base	236M	96.88	81.60
XLM-R Base	278M	95.84	79.48
XLM-V	780M	44.68	78.50
IndicBERT-v2-SS	278M	38.20	74.03
IndicBERT-v2-MLM-only	278M	98.00	83.77
IndicBERT-v2-MLM-Back-TLM	278M	98.03	84.00
IndicBERT-v2-MLM-Sam-TLM	278M	97.69	83.72
TamilBERT	237M	97.89	83.35
Tamil-SQuAD-BERT	237M	97.42	83.41
Indic-TamArjun (Ours)	**44M**	**98.15**	**90.08**

Table 5: Performance comparison (Macro-F1 scores) of our proposed Indic-TamArjun model with state-of-the-art models across Tamil language tasks. **Bold** indicates results that are better with statistical significance ($p < 0.005$).

derperform on specific tasks, reinforcing our finding that parameter efficiency combined with a well-designed multi-objective pre-training methodology can outperform much larger models. The strong performance across diverse tasks highlights how our joint training methodology helps develop more robust and transferable representations.

4.5.3 Ablation Studies

To investigate the individual and combined effects of different pre-training objectives, we conducted an ablation study with various combinations of tasks. From the 13 possible combinations of objectives, we selected three single-task configurations (MT, MS, and CS) that utilize the encoder, one two-task combination (MT+MS), and one three-task combination (MT+MS+NT) to compare against our full model. Table 7 and Table 8 present the results for Bangla and Tamil respectively.

The results demonstrate a clear trend: models pre-trained with more objectives consistently outperform those trained with fewer objectives across most tasks. For both languages, the full models with all four pre-training objectives (Vac-Arjun and Indic-TamArjun) achieve the best performance on all downstream tasks. This confirms our hypothesis that the combination of MT, NT, MS, and CS objectives creates synergistic effects by addressing complementary aspects of language understanding. Just as human language proficiency requires mastery of multiple skills—vocabulary (MT), grammar and flow (NT), phrase composition (MS), and discourse coherence (CS)—models trained on diverse objectives develop richer linguistic competence than single-task approaches. MT provides bidirectional contextual understanding through rich embeddings, NT teaches sequential generation patterns and causal dependencies, MS captures multi-token longer-range semantic dependencies, and CS develops sentence-level coherence given partial inputs. This multi-faceted training mirrors how humans naturally acquire comprehensive language knowledge through exposure to various linguistic tasks, resulting in models that excel at both understanding and generation without the limitations of single-objective frameworks. Interestingly, even though our downstream evaluations focus solely

Model	Parameters	Tasks				
		BBC News Classification	NER	IIT Patna Movie Review	IIT Patna Product Review	MIDAS Discourse
mBERT	180M	38.03	84.11	62.13	71.21	59.63
MuRIL Base	236M	40.76	89.49	68.13	78.05	60.94
MuRIL Large	505M	28.86	89.58	67.51	78.76	9.20
XLM-R Base	278M	42.33	87.48	63.70	75.65	65.97
XLM-V	780M	40.29	86.85	18.83	40.12	57.34
IndicBERT-v2-SS	278M	50.38	87.29	70.63	78.61	63.33
IndicBERT-v2-MLM-only	278M	55.79	90.31	70.60	78.86	66.04
IndicBERT-v2-MLM-Back-TLM	278M	52.72	89.85	**71.29**	78.96	66.38
IndicBERT-v2-MLM-Sam-TLM	278M	55.25	90.49	71.25	**79.37**	66.16
HindiBERT-v2	237M	28.88	89.43	71.04	76.41	61.32
Hindi-SQuAD-BERT	237M	47.03	88.86	66.18	76.94	65.17
Indic-HindArjun (Ours)	**44M**	**62.06**	**91.39**	71.16	79.06	**69.41**

Table 6: Performance comparison (Macro-F1 scores) of our proposed Indic-HindArjun model with state-of-the-art models across Hindi language tasks. **Bold** Indicates results that are better with statistical significance ($p < 0.005$).

on encoder-based tasks, incorporating the NT objective (which is decoder-only during pre-training) still improves the encoder's performance. This suggests that decoder training creates a beneficial effect on the shared embedding layer and indirectly influences the encoder weights through the joint optimization process.

It's important to note that while incorporating multiple objectives improves model performance, it also increases pre-training time as shown in the tables, approximately 2.8x slower when using all four objectives compared to using just MT. However, this is only a pre-training cost, as inference time remains the same across all variants since the model architecture and parameters are same irrespective of the number of pre-training tasks.

5 Conclusions and Future Directions

In this work, we have presented a novel multi-objective pre-training framework that strategically combines four complementary objectives to develop parameter-efficient language models. Our Arjun model family, with only 44M parameters for downstream tasks, consistently achieves competitive or superior performance compared to models that are 2.5-12.5 times larger across 12 diverse tasks in Bangla, Tamil, and Hindi. The ablation studies confirm that joint optimization of all four objectives creates synergistic effects that outperform single-objective or partial combinations, validating our hypothesis about the complementary nature of bidirectional understanding and autoregressive modeling within a unified framework.

This work opens several research avenues. First, evaluating our models on generative tasks using the decoder component will provide insights into the full potential of Arjun. Second, incorporating additional pre-training objectives such as ELECTRA's replaced token detection further enhance representation quality. Third, extending our approach to more languages in monolingual as well as multilingual settings will demonstrate the broader applicability of our multi-objective framework.

Model	Sentiment 2-class	Sentiment 3-class	Poem	NER	Spell Error	Pre-training Time (sec/step)
MT	94.44	69.08	63.19	92.27	81.90	0.50
MS	94.25	69.05	61.79	92.25	80.65	0.67
CS	94.24	69.40	62.90	92.18	82.12	0.67
MS + MT	94.64	69.61	64.01	92.11	82.22	0.88
MS + MT + NT	94.50	69.13	64.40	92.45	82.25	1.17
Vac-Arjun (All 4 tasks)	**95.17**	**70.06**	**65.23**	**93.57**	**82.45**	1.38

Table 7: Ablation study results (Macro-F1 scores) on Bangla tasks with different pre-training objective combinations. All models have 44M parameters.

Model	News Category	NER	Pre-training Time (sec/step)
MT	98.03	87.12	0.51
MS	97.55	88.69	0.66
CS	97.78	87.91	0.66
MS + MT	98.07	88.73	0.88
MS + MT + NT	98.00	89.34	1.18
Indic-TamArjun (All 4 tasks)	**98.15**	**90.08**	1.37

Table 8: Ablation study results (Macro-F1 scores) on Tamil tasks with different pre-training objective combinations. All models have 44M parameters.

Acknowledgements

Following ARR's AI Writing/Coding Assistance Policy, we acknowledge using Claude 3.7 Sonnet (Anthropic, 2025) for editorial assistance while maintaining full ownership of all scientific contributions.

6 Limitations

While our multi-objective pre-training approach demonstrates promising results for parameter-efficient Indic language models, several limitations should be acknowledged. Our evaluation focuses on only three Indian languages (Bangla, Tamil, and Hindi) out of the hundreds of languages spoken in the Indian subcontinent. This limited scope may not fully represent the linguistic diversity and complexity of the broader Indian language family. The effectiveness of our approach on morphologically richer languages, tonal languages, or languages with different writing systems within the Indian family remains to be empirically validated.

Our evaluation is limited to discriminative tasks such as sentiment analysis, named entity recognition, and classification. We have not evaluated the generative capabilities of our encoder-decoder models using the decoder component, which could provide insights into the full potential of our multi-objective pre-training framework. The performance on tasks requiring complex reasoning, creative generation, or long-form text production remains unexplored.

While we train separate monolingual models, we have not thoroughly investigated the cross-lingual transfer capabilities of our approach. The potential for knowledge sharing across related Indian languages through multilingual training or the development of universal representations remains unexplored. Our evaluation relies primarily on existing benchmark datasets, which may not capture the full range of linguistic phenomena or real-world applications relevant to Indian languages. The relatively small size of some evaluation datasets may limit the statistical significance of our comparisons, particularly for specialized tasks like poem classification.

7 Ethics Statement

The development and deployment of language models for Indian languages raises several important ethical considerations. Our models are trained on text corpora that may not equally represent all speakers of Bangla, Tamil, and Hindi. The Vācaspati corpus focuses on literary texts, while IndicCorp v2 primarily contains web-scraped content, potentially underrepresenting certain demographic groups, dialects, or socioeconomic backgrounds. We acknowledge that model performance may vary across different user groups, and we encourage future work to evaluate fairness across diverse populations within each language community.

Working with Indian languages requires careful consideration of cultural nuances, historical context, and community values. We recognize that our computational approach may not capture the full cultural richness embedded in these languages. We have made efforts to use established, community-validated datasets and evaluation benchmarks where possible, but acknowledge that automated systems may inadvertently perpetuate or amplify existing societal biases present in training data.

Training multiple language-specific models requires significant computational resources, contributing to carbon emissions and environmental impact. While our parameter-efficient approach reduces these costs compared to larger models, we acknowledge the environmental responsibility associated with computational research. We recommend that future applications of this work consider the environmental trade-offs and explore carbon offset options where possible.

The training corpora used in this work were collected from various sources, including web-scraped content. While we relied on established, publicly available datasets, we recognize the importance of data consent and privacy. Future work should prioritize the development of ethically sourced datasets with explicit consent from content creators and communities. Language models can potentially be misused for generating harmful content, spreading misinformation, or automating deceptive practices. While our evaluation focuses on beneficial applications, we acknowledge that our models could theoretically be applied to harmful uses. We encourage responsible deployment and recommend implementing appropriate safeguards and use-case restrictions in practical applications.

While our work aims to democratize NLP technology for Indian languages, the computational requirements may still create barriers for researchers and practitioners in resource-constrained environments. We are committed to making our models and findings publicly available to promote equitable access to this technology. We recognize that meaningful progress in Indian language technology requires ongoing engagement with native speaker communities, linguists, and cultural experts. Our current work represents a computational contribution, but we acknowledge the importance of incorporating community feedback and ensuring

that technological advances align with community needs and values. We commit to providing detailed documentation of our methods, hyperparameters, and evaluation procedures to ensure reproducibility and plan to release our trained models and code to enable further research and community validation of our findings.

References

Anthropic. 2025. Claude 3.7 sonnet [large language model]. https://www.anthropic.com. Released February 24, 2025. Accessed 2025-05-19.

Abhik Bhattacharjee, Tahmid Hasan, Kazi Samin, M Sohel Rahman, Anindya Iqbal, and Rifat Shahriyar. 2022. Banglanlp: Tools and datasets for bengali natural language processing. *arXiv preprint arXiv:2203.15065*.

Pramit Bhattacharyya, Joydeep Mondal, Subhadip Maji, and Arnab Bhattacharya. 2023. Vacaspati: A diverse corpus of bangla literature. *arXiv preprint arXiv:2307.05083*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Swapnil Dhanwal, Hritwik Dutta, Hitesh Nankani, Nilay Shrivastava, Yaman Kumar, Junyi Jessy Li, Debanjan Mahata, Rakesh Gosangi, Haimin Zhang, Rajiv Ratn Shah, and Amanda Stent. 2020. An annotated dataset of discourse modes in hindi stories. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 1191–1196, Marseille, France. European Language Resources Association.

Michael Diskin, Aleksey Bukhtiyarov, Max Ginsburg, Jai Patel, and Christian Egele. 2021. Distributed training of bert-like models across multiple tpu nodes. In *International Conference on Machine Learning Hardware-Aware Efficient Training Workshop*.

Sumanth Doddapaneni, Rahul Aralikatte, Gowtham Ramesh, Shreya Goyal, Mitesh M. Khapra, Anoop Kunchukuttan, and Pratyush Kumar. 2022. Towards Leaving No Indic Language Behind: Building Monolingual Corpora, Benchmark and Models for Indic Languages. *arXiv e-prints*, arXiv:2212.05409.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Khondoker Ittehadul Islam, Md Saiful Islam, Sudipta Kar, and Mohammad Ruhul Amin. 2021. Sentnob: A dataset for analysing sentiment on noisy bangla texts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Raviraj Joshi. 2022. L3cube-hindbert and devbert: Pre-trained bert transformer models for devanagari based hindi and marathi languages. *arXiv preprint arXiv:2211.11418*.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, NC Gokul, Avik Bhattacharyya, Mitesh M Khapra, and Pratyush Kumar. 2020. Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961.

Ke Ke, Hiroyuki Shindo, and Yuji Matsumoto. 2024. Spactor-t5: Pre-training t5 models with span corruption and token replacement. *arXiv preprint arXiv:2401.13160*.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, and 1 others. 2021. Muril: Multilingual representations for indian languages. *arXiv preprint arXiv:2103.10730*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Crosslingual name tagging and linking for 282 languages. In

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

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.

Ofir Press and Lior Wolf. 2016. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Salim Sazzed. 2020. Cross-lingual sentiment classification in low-resource bengali language. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 50–60, Online. Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, and 1 others. 2022. Ul2: Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131*.

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

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, 32.

This is an appendix.