Efficient Transformer-Based Abstractive Urdu Text Summarization Through Selective Attention Pruning

Muhammad Azhar * , Adeen Amjad , Ghulam Farid , Deshinta Arrova Dewi , Malathy Batumalay

Posted Date: 11 November 2025

doi: 10.20944/preprints202511.0601.v1

Keywords: automatic text summarization; abstractive; urdu language; transformer; attention pruning; process optimization

 doi:10.20944/preprints202511.0601.v1

Article

Efficient Transformer-Based Abstractive Urdu Text Summarization Through Selective Attention Pruning

Muhammad Azhar [1,*], Adeen Amjad [2], Ghulam Farid [3], Deshinta Arrova Dewi [4] and Malathy Batumalay [4]

[1] Department of Applied Data Science, Hong Kong Shue Yan University, Hong Kong SAR, China
[2] Department of Computer Science, University of Sahiwal, Pakistan
[3] College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin 150001, China
[4] Faculty of Data Science and Information Technology, INTI International University, Nilai 71800, Negeri Sembilan, Malaysia
* Correspondence: azhar@hksyu.edu

Abstract

In today's data-driven world, automatic text summarization is essential for extracting insights from large data volumes. While extractive summarization is well-studied, abstractive summarization remains limited, especially for low-resource languages like Urdu. This study introduces process innovation through transformer-based models—Efficient-BART (EBART), Efficient-T5 (ET5), and Efficient-GPT-2 (EGPT-2) optimized for Urdu abstractive summarization. Innovations include strategically removing inefficient attention heads to reduce computational complexity and improve accuracy. Theoretically, this pruning preserves structural integrity by retaining heads that capture diverse linguistic features, while eliminating redundant ones. Adapted from BART, T5, and GPT-2, these optimized models significantly outperform their originals in ROUGE evaluations, demonstrating the effectiveness of process innovation and optimization for Urdu natural language processing.

Keywords: automatic text summarization; abstractive; urdu language; transformer; attention pruning; process optimization

1. Introduction

As the digital landscape continues to evolve, the exponential growth of textual data presents both significant opportunities and challenges in information management [1]. The complexity of modern information systems extends beyond single-language, single-domain contexts to encompass multilingual, multi-modal, and cross-jurisdictional data, such as in comprehensive coastal zone knowledge systems [2]. The vast amount of information spanning various domains has become increasingly difficult to navigate, whether in academic publications, news articles, or organizational reports. Innovative solutions are urgently needed to streamline information retrieval and comprehension, enabling efficient extraction of relevant insights from extensive document collections.

Automatic text summarization has emerged as a critical technology to address these challenges by condensing lengthy documents into concise and informative summaries. By providing users with quick access to key points and essential elements, text summarization facilitates more efficient decision-making, knowledge acquisition, and information management processes.

Text summarization approaches are generally categorized into two main types: extractive and abstractive methods [3,4]. Extractive summarization identifies and concatenates the most relevant sentences from the original text, while abstractive summarization attempts to understand underlying meanings and generate novel summaries using techniques such as paraphrasing and rewriting, resulting in more human-like outputs. Urdu, as a linguistically rich language with complex morphology, right-to-left script, and significant morphological complexity, presents unique challenges for text summarization that differ substantially from English and other high-resource languages [5,6]. The language's compound morphology, lack of word boundaries, and limited availability of annotated

 doi:10.20944/preprints202511.0601.v1

datasets for model training and evaluation create substantial barriers to effective NLP application development.

This study addresses the following research questions:

1. Can selective attention head pruning improve the performance and efficiency of transformer models for Urdu abstractive summarization?
2. How do different transformer architectures (BART, T5, GPT-2) respond to attention head pruning when adapted for Urdu?
3. What is the optimal pruning threshold that balances performance and efficiency for Urdu text summarization?
4. How do the optimized models generalize across different domains of Urdu text?

Transformer-based architectures, such as BART, T5, and GPT-2, have set new benchmarks in natural language processing tasks, including summarization for high-resource languages [7]. We selected these three models specifically because they represent distinct architectural paradigms: BART uses a bidirectional encoder and autoregressive decoder, T5 employs a text-to-text unified framework, and GPT-2 utilizes a decoder-only autoregressive approach. This diversity allows for a comprehensive evaluation of our pruning methodology across different transformer designs. While newer models like GPT-3 and GPT-4 exist, we chose GPT-2 for its open-source nature, reproducibility, and suitability for architectural modification experiments, focusing on process innovation rather than scale-based performance gains. Their application to Urdu, however, reveals a critical research gap. While current state-of-the-art approaches for Urdu typically involve fine-tuning these pre-trained models, they operate under a significant limitation: they apply the full, unmodified transformer architecture to a specialized, low-resource task. These architectures contain attention heads trained on a mixture of languages and tasks, many of which become redundant or inefficient when specialized for Urdu abstractive summarization. This results in models that are computationally expensive, sub-optimally efficient, and not tailored to the specific linguistic characteristics of Urdu. Merely fine-tuning pre-trained models without architectural optimization fails to address the inherent inefficiencies for a targeted, low-resource application.

To bridge this gap, this study introduces a novel optimization approach that moves beyond simple fine-tuning. We posit that not all components of large transformer architectures contribute equally to the task of Urdu abstractive summarization. By strategically identifying and removing inefficient attention heads—a process we term selective attention pruning we create leaner yet more effective models. This approach is grounded in the observation that attention heads vary in their contribution; eliminating redundant heads can significantly reduce computational complexity while maintaining or even enhancing summarization quality. Building on this principle, we develop and evaluate three optimized models: Efficient-BART (EBART), Efficient-T5 (ET5), and Efficient-GPT-2 (EGPT-2).

The key research contributions of this paper are:

- Development of optimized transformer models (EBART, ET5, and EGPT-2) through strategic attention head pruning, specifically designed for Urdu abstractive summarization;
- Comprehensive investigation of leading transformer architectures (BART, T5, and GPT-2) for Urdu language processing;
- Extensive evaluation of the optimized models using ROUGE metrics and comparative analysis against their original counterparts.

The remainder of this paper is organized as follows. Section 2 discusses related work in text summarization. Section 3 provides detailed descriptions of datasets, preprocessing methods, and model architectures. Experimental setup, results in Section 4 , and discussions are presented in Section . Finally, Section 6 concludes the study and outlines future research directions.

 doi:10.20944/preprints202511.0601.v1

2. Background and Literature Review

This section examines the technical foundations of automatic text summarization. It provides a comprehensive review of current state-of-the-art approaches, with particular emphasis on transformer-based methods and their applications to low-resource languages.

Egonmwan and Chali [8] proposed transformer-based models for single-document neural summarization. Their work showed that sequence-to-sequence models with attention can outperform earlier systems but require longer training time.

Building on this work, Abolghasemi et al. [9] developed HTS-DL, a hybrid text summarization system that combines extractive and abstractive summarizers to overcome the challenges of traditional neural models. Their approach outperformed existing models by leveraging the strengths of both summarization paradigms

2.1. Single Document Summarization Approaches

Single-document summarization has improved significantly with deep learning models. Jiang et al. [10] introduced attention-based bidirectional LSTMs along with sequence-to-sequence long short-term memory networks to handle problems like out-of-vocabulary words and redundant outputs. Experiments on public corpora showed that their system outperformed baseline and multiple state-of-the-art systems. Lewis et al.'s [7] introduction of BART additionally improved abstractive summarization using a bidirectional encoder and autoregressive decoder, with a two-stage pretraining strategy (corruption and reconstruction of text), allowing very effective end-to-end summarization. While BERT-based approaches have shown promise for text understanding tasks in various languages [11,12]. Morphologically rich languages present special difficulties for summarizing text, however. While in general, summarization algorithms are language-independent, languages like Urdu need custom preprocessing. Daud et al. [5] stressed that the compound morphology and lack of word boundaries in Urdu make it important to develop language-specific tools like stemmers, lemmatizers, and stopword lists. Farooq et al. [13] solved these problems by comparing different summarization methods for Urdu. In contrast, Asif et al. [14] presented a hybrid architecture combining TF–IDF weighting, word frequency, and transformer-based abstractive generation. Their architecture generated summaries of the same quality as human-generated text, with better coherence and retrieval utility.

2.2. Multiple Document Summarization and Hybrid Approaches

Hybrid models combining several approaches have reached competitive performance in multi-document summarization. Khyat et al. [15] introduced an N-gram-based hybrid model with deep learning, which outperformed traditional systems with better ROUGE scores. Mujahid et al. [16] improved transformer architectures for Urdu headline classification, demonstrating that fine-tuning over Urdu corpora results in significant performance gains. Their results established that transformer-based features support effective text representation and categorization for Urdu language processing.

Transformer architectures revolutionized natural language processing. Vaswani et al. [17] proposed the self-attention mechanism that lies at the heart of all modern transformer models. Lin et al. [18] provided the ROUGE metric for automatic evaluation of summaries, and their framework became the standard evaluation platform for summarization-based tasks in text processing. Finally, Wolf et al. [19] developed the Transformers library that unified BERT, GPT, and T5 architectures for a wide variety of NLP tasks and significantly accelerated both research and deployment.

Recent works have been done on optimizing transformer architectures by analyzing and pruning components. Michel et al. [20] examined attention head redundancy and asked the question if sixteen heads are really better than one. This has been a path towards efficient architecture design. Cheema et al. [21] showed that selective pruning of attention heads alone could achieve substantial gains in efficiency for GPT-2 and established a method for optimizing transformers through selective reduction of components.

 doi:10.20944/preprints202511.0601.v1

A few approaches have emerged in the case of low-resource language scenarios. Savelieva et al. [22] conducted abstractive summarization of instructional texts using BERT, confirming its context understanding capabilities. Xue et al. [23] proposed mT5, a multilingual text-to-text transformer, showing great robustness for multilingual and low-resource summarization tasks. Munaf et al. [24] focused on low-resource summarization with pre-trained language models and depicted good results using these models in a low-resource setting.

Cross-lingual adaptations have achieved considerable success in Urdu language processing. Khalid et al. [25] developed RUBERT, a Roman Urdu BERT bilingual model. Extensive social media data was used to develop it and enhance the understanding of Urdu text. Rauf et al. [26] explored the application of deep learning methods to fake news detection in Urdu. This demonstrates the potential of transformer-based techniques for Urdu NLP tasks.

Azhar et al. [27] conducted a comprehensive systematic review and experimental evaluation of classical and transformer-based models for Urdu abstractive text summarization. Their work benchmarked various architectural approaches and identified the unique challenges posed by Urdu's linguistic characteristics in low-resource settings, providing crucial insights into the current state of Urdu summarization research.

Despite these advancements, significant limitations persist in Urdu text summarization. Current state-of-the-art models predominantly rely on fine-tuning pre-trained multilingual transformers without structural modifications. This approach leaves many attention heads redundant or inefficient when specialized for Urdu, resulting in unnecessary computational overhead and suboptimal performance. To address this gap, our work introduces a selective attention head pruning mechanism that optimizes transformer architecture at the structural level, moving beyond parameter adjustment through fine-tuning. This architectural optimization enables the development of faster, lighter, and more efficient transformer models specifically engineered for abstractive Urdu text summarization tasks.

3. Methodology

This section explains the steps of our proposed methodology. First, the data preprocessing steps used for our proposed techniques are discussed. Then, the process of strategic removal of inefficient heads is discussed, which is used in all three optimized versions, i.e., EBART, ET5, and EGPT-2, to improve Urdu abstractive text summarization.

3.1. Dataset Description and Preprocessing

The **Urdu Fake News Dataset** from Hugging Face was used for this study. It consists of a total of **1,300 news articles** collected from authentic Urdu news sources. The dataset is categorized into five domains: Business, Health, Showbiz, Sports, and Technology. The distribution of real and fake articles across these categories is as follows: the Business and Sports categories each contain 150 real and 80 fake articles; the Health, Showbiz, and Technology categories each contain 150 real and 130 fake articles. This results in a total of 750 real and 550 fake articles, providing a robust foundation for model training and evaluation.

The dataset was split into 80% for training (1,040 articles) and 20% for testing (260 articles). Each article is paired with a human-written abstractive summary, which serves as the ground truth for the summarization task.

Before training and evaluating the proposed models, several pre-processing steps were performed on the Urdu text. Given Urdu's unique linguistic characteristics, including right-to-left script, complex morphology, and lack of clear word boundaries [5], the transformer architectures (GPT-2, T5, and BART) are designed to handle raw text input directly, leveraging knowledge acquired during pre-training. This makes the data preprocessing steps relatively straightforward.

The following pre-processing pipeline was applied consistently to all models to ensure experimental consistency and fair comparison:

 doi:10.20944/preprints202511.0601.v1

1. Tokenization: The input Urdu text was tokenized into subword tokens using the respective pre-trained tokenizers for each model (e.g., `GPT2Tokenizer`, `T5Tokenizer`, `BartTokenizer`).[25]
2. Sequence Length Adjustment: The tokenized sequences were padded or truncated to a fixed length of 512 tokens to match the models' expected input dimensions.
3. Text Normalization: Basic normalization was applied, including converting text to lowercase and removing extraneous punctuation marks.
4. Tensor Conversion: The final tokenized and adjusted sequences were converted into PyTorch tensors, the required format for model inference and fine-tuning.

This standardized pre-processing ensures that the models receive clean, uniformly formatted input, enabling effective learning and generalization.

3.2. Efficient Transformer Summarizer Models by Pruning Attention Heads Based on Their Contribution

This section provides a detailed explanation of the efficient transformer summarizer framework and the pruning process of attention heads based on their contribution. The pre-processed Urdu text dataset is used as the input for the model's training. The complete framework of the proposed efficient transformer summarizer is illustrated in Figure 1. Each step of the proposed framework is discussed step by step.

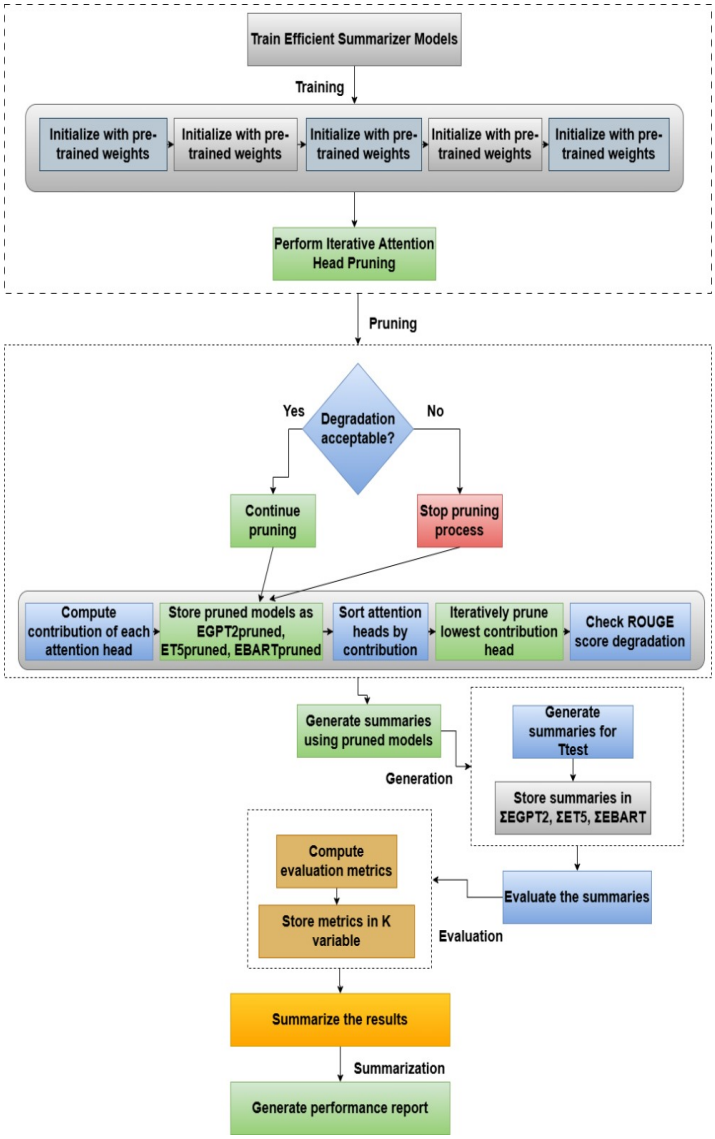

Figure 1. The proposed framework for training and evaluating efficient Transformer summarizers (EBART, ET5, EGPT-2).

 doi:10.20944/preprints202511.0601.v1

3.2.1. Training the Efficient Summarizer Models

The training process described in this section is applied to all three Efficient Transformer Summarizer models: 1) Efficient GPT-2 (EGPT-2), 2) Efficient T5 (ET5), and 3) Efficient BART (EBART). Each model is initialized with the pre-trained weights of its original counterpart (GPT-2, T5, and BART, respectively) and is then fine-tuned on the T_{train} dataset containing the tokenized Urdu training documents. The overall procedure for the Efficient Transformer Summarizer is outlined in Algorithm 1.

The training and optimization pipeline consists of three major phases designed to collectively enhance performance and efficiency:

1. **Fine-tuning:** This phase adapts the pre-trained models to the specific domain of Urdu text summarization, exposing them to Urdu language patterns and content features [7]. This foundational step allows the model to learn Urdu-specific linguistic structures and contextual dependencies. The computational complexity of this phase is $O(N \cdot L^2)$ per epoch, where N is the number of training samples and L is the sequence length.
2. **Pruning:** This phase systematically removes redundant attention heads based on their contribution to the ROUGE score, reducing model complexity and sharpening focus on salient Urdu linguistic features. By eliminating heads that contribute minimally to the task, the model becomes more efficient and focused. The iterative pruning process has a complexity of $O(H \cdot E)$, where H is the number of attention heads and E is the evaluation cost per head.
3. **Evaluation:** This phase ensures that pruning does not lead to performance degradation, guaranteeing the final model is both efficient and accurate. This validation step confirms that the pruned model retains its ability to generate high-quality Urdu summaries while achieving computational gains.

A critical component of the fine-tuning process is hyperparameter optimization. The specific values used in our experiments, which were determined through empirical validation on a held-out development set, are as follows:

- **Number of encoder/decoder layers:** 6 (for BART and T5), 12 (for GPT-2).
- **Number of attention heads:** 12 (for BART and T5), 12 (for GPT-2).
- **Embedding size:** 768.
- **Feed-forward network size:** 3072.
- **Dropout rate:** 0.1.
- **Learning rate:** 1e-4, using the AdamW optimizer.
- **Batch size:** 8.
- **Sequence length:** 512 tokens.
- **Weight decay:** 0.01.
- **Training epochs:** 200

In addition to this, a linear warm-up schedule was employed to gradually increase the learning rate before reaching the maximum limit for optimization stability. The weight decay of L2 regularization, applied to the model weights, also helps prevent overfitting, which is crucial for better training and fine-tuning processes. Training was conducted for 200 epochs or until early stopping was triggered based on performance on a held-out validation set. After the fine-tuning process, the EBART, ET5, and EGPT-2 models are further utilized for the pruning of attention heads, which is explained in the next section.

 doi:10.20944/preprints202511.0601.v1

Algorithm 1 Pruning and Summarization with Efficient GPT-2, T5, and BART Models

Require:
Train: Tokenized training text documents
Test: Tokenized testing text documents
Tpruned: Tokenized pruning text documents
θ: Accuracy sliding window for attention score pruning
Ensure:
Σ_{EGPT-2}: Summaries by pruned Efficient GPT-2 model
Σ_{ET5}: Summaries by pruned Efficient T5 model
Σ_{EBART}: Summaries by pruned Efficient BART model
K: Evaluation metrics [F1-score, Precision, Recall, BLEU, ROUGE]
`EGPT-2_Pruned`, `ET5_Pruned`, `EBART_Pruned`: Pruned summarizer models
procedure MAIN ALGORITHM
    # Train the Efficient Summarizer models
    $EGPT2 \leftarrow$ TRAINEGPT2(T_{train}, GPT-2)
    $ET5 \leftarrow$ TRAINET5(T_{train}, T5)
    $EBART \leftarrow$ TRAINEBART(T_{train}, BART)
    # Perform Iterative Attention Head Pruning
    `EGPT-2_Pruned` $\leftarrow$ PERFORM_PRUNING($EGPT2$, T_{pruned}, θ)
    `ET5_Pruned` $\leftarrow$ PERFORM_PRUNING($ET5$, T_{pruned}, θ)
    `EBART_Pruned` $\leftarrow$ PERFORM_PRUNING($EBART$, T_{pruned}, θ)
    # Generate summaries using pruned models
    $\Sigma_{EGPT-2} \leftarrow$ GENERATE_SUMMARIES(`EGPT-2_Pruned`, T_{test})
    $\Sigma_{ET5} \leftarrow$ GENERATE_SUMMARIES(`ET5_Pruned`, T_{test})
    $\Sigma_{EBART} \leftarrow$ GENERATE_SUMMARIES(`EBART_Pruned`, T_{test})
    # Evaluate the summaries
    $K \leftarrow$ EVALUATE_SUMMARIES($\Sigma_{EGPT-2}, \Sigma_{ET5}, \Sigma_{EBART}$, $testGround\_Truth$)
    # Summarize the results
    SUMMARIZE_RESULTS(K)
end procedure

GPT-2, T5, and BART are fine-tuned on the T_{train} dataset, and the process starts from the pre-trained weights of the GPT-2, T5, and BART models to do Urdu text summarization. During the fine-tuning process on the T_{train} dataset, the language patterns, content features, and specific patterns present in the training corpus are exposed to the models, enabling them to fine-tune their parameters and internal representations to reflect the subtleties of the target domain accurately.

The key step of the fine-tuning process is hyperparameter tuning, like

- The optimal number of transformer blocks or layers in the encoder and decoder part of the transformer to avoid overfitting and better generalization.
- Number of attention heads in the multi-head attention mechanism to optimally focus on different parts of the input text simultaneously.
- Optimal embedding size to capture more information with optimal computational resources.
- Optimal feed-forward neural network size within each transformer block to increase the model's capacity and decrease the complexity.
- Optimal dropout rate for various transformer layers to prevent overfitting by randomly dropping out some of the units during training.
- Optimal learning rate used by the optimization algorithm, like Adam, SGD, etc., during the fine-tuning process to converge quickly.
- Optimal batch size of the number of paragraphs/sentences processed in a single forward and backward pass during training for more stable gradients.
- Optimal sequence length of the input and output sequences to capture more context and optimal usage of computational resources, etc.

In addition to this, warm-up steps are used to gradually increase the learning rate before reaching the maximum limit for optimization stability. The weight decay of L2 regularization, applied to the

 doi:10.20944/preprints202511.0601.v1

model weights, also helps prevent overfitting, which is crucial for better training and fine-tuning processes. Usually, training is continued for a specified number of epochs or until the models converge (i.e., no longer improvement in the performance on a held-out validation set). After the fine-tuning process, the EBART, ET5, and EGPT-2 models are further utilized for the pruning of attention heads, which is explained in the next step.

3.2.2. Perform Iterative Attention Heads Pruning with Individual Contribution Computation

Iterative attention head pruning based on individual contribution is the key step of our efficient summarizer models. Each of the trained Efficient Summarizer models undergoes an iterative attention-head pruning process in the next phase of the fine-tuning process.
The function `perform_iterative_pruning(model, Tpruned, θ)`,
where model $\in$ {EGPT2, ET5, EBART}, is responsible for performing the iterative attention heads pruning based on the individual contribution of the attention heads. EGPT2, ET5, and EBART models, along with the pruned tokenized text documents T_{pruned} and an accuracy sliding window threshold θ, are input to the iterative attention head pruning function.

To quantify the contribution of each attention head to the performance of the model, the ROUGE score [21] is employed as the measure. We use ROUGE as our primary performance metric to quantify "accuracy" in the context of summarization quality. The contribution of each attention head is measured by computing the drop in ROUGE score when that specific head is masked—a significant drop indicates a high-contribution head, while a minimal drop suggests redundancy [27]. The function measures the contribution of each attention head and ranks them from low to high based on their contribution. The ROUGE score guarantees accuracy with the assistance of the accuracy sliding window threshold *theta*.

To determine the individual contribution of a particular attention head, it is masked to see how well the model performs without that particular attention head. This process is performed for each attention head to measure its contribution during each iteration. Let M denote the original model and $M^{(-i)}$ the model with the i-th attention head removed or masked.

The importance of the attention head can be quantified as:

$$\text{Importance}(i) = \text{Performance}(M) - \text{Performance}(M^{(-i)}) \tag{1}$$

where Performance $(\cdot)$ is a chosen metric, such as summarization quality, perplexity, or other relevant metrics. We use the ROUGE score to check the performance of the text summarization.

Through the iterative pruning of attention heads, redundant and inefficient attention heads are removed, enabling a higher level of model compression. By pruning the attention heads that have no significant influence on the prediction, we ensure an optimal Transformer Summarizer that achieves better accuracy with optimal resource utilization by reducing the complexity of the base model. Next, the pruned models are stored as `EGPT-2_Pruned`, `ET5_Pruned`, and `EBART_Pruned`, which correspond to the pruned versions of the Efficient-GPT-2 (EGPT-2), Efficient-T5 (ET5), and Efficient-BART (EBART) models, respectively.

Visualizing Pruning Effects: Figure 2 illustrates the transformative effect of selective attention head pruning on inter-layer information flow. The pre-pruning configuration (left) maintains all 12 attention heads, creating redundant computational pathways. In contrast, the optimized post-pruning architecture (right) retains only 8 high-contribution heads while eliminating 4 low-contribution heads (H3, H6, H8, H10), resulting in focused processing of Urdu linguistic patterns. This architectural optimization directly addresses Urdu's unique morphological challenges by concentrating computational resources on heads that capture language-specific features, while eliminating redundant heads trained on cross-linguistic patterns irrelevant to Urdu summarization. The detailed process of pruning is shown in Figure 3 and Algorithm 2. Algorithm 2 describes the entire procedure of Attention Head pruning based on the individual contribution in the multi-head attention part. This algorithm shows the functionality of the function `perform_iterative_pruning(model, Tpruned, θ)`, where

 doi:10.20944/preprints202511.0601.v1

model $\in$ {EGPT2, ET5, EBART} is used in all three proposed models, i.e., EGPT2, ET5, and EBART. M and H are given as input to Algorithm 2, where M is the original model, and H is the total number of attention heads in the model. MP and CP are the outputs of Algorithm 2, where MP is the optimal model consisting of the high-performing attention heads after the removal of the least contributing attention heads, while CP is the contribution of each attention head in the optimal model.

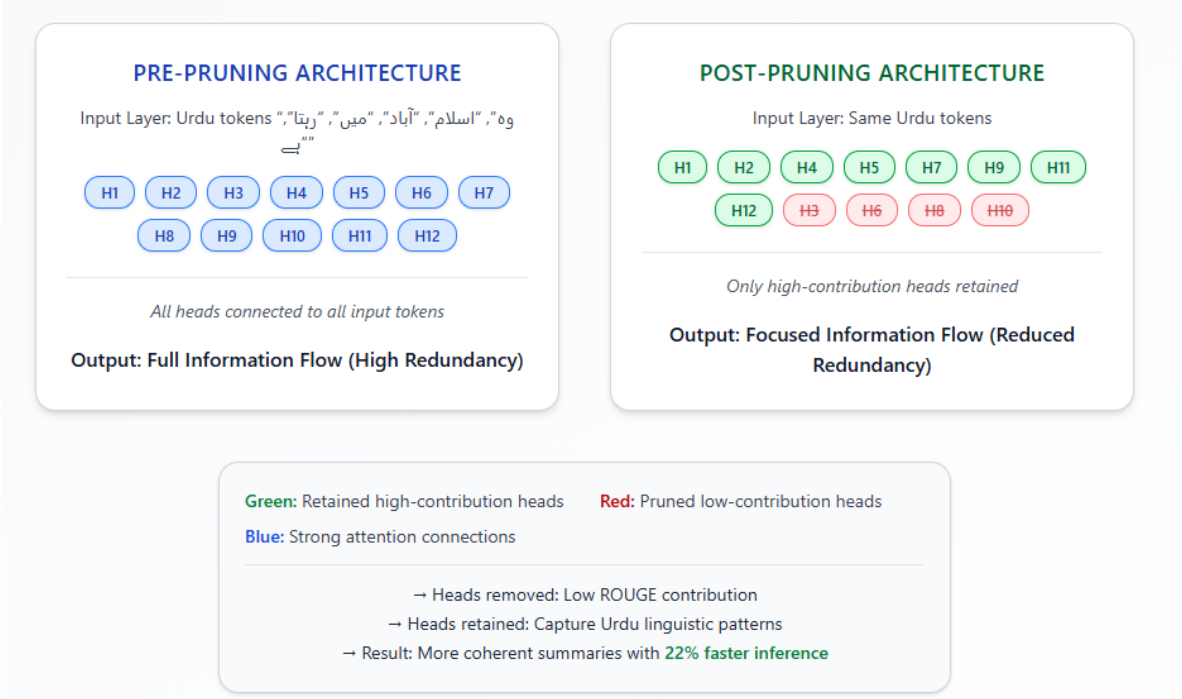

Figure 2. Schematic of Inter-Layer Information Flow Before and After Attention Head Pruning.

In the standard model M, the function `ComputeAttentionHeadContribution (M, H)` is used to obtain the contribution of each attention head H. The C is the vector with the contribution of each attention head. At the beginning of the pruning process, the original model M is set as the trimmed model MP, and the original contribution vector C is assigned to the contribution vector CP. The list hr is established to keep track of the removed heads. In each iteration, h_{min} is computed, which is the index of the attention head that contributes the least in CP. The function `RemoveAttentionHead(MP, hmin)` removes the h_{min}-th attention head from the MP, resulting in a new model MP/h_{min}, which is the model without the attention head of the minimum contribution.

The functions `ComputeROUGE(MP)` and `ComputeROUGE(MP/hmin)` are used to calculate the ROUGE scores for the current model MP and the pruned model MP/h_{min} (a model after removing the attention head with a minimum individual contribution based on the ROUGE scores), respectively. The function `ComputeAttentionHeadContribution(MP, HP)` again computes the new contribution of each attention head in the pruned model MP, finds the index h_{min} of the attention head with the minimum contribution in CP, and removes the inefficient attention head with a minimum contribution. This process continues until the accuracy increases or equals the previous accuracy of the model. The loop breaks if the accuracy starts decreasing. The program returns the optimal pruned model MP together with the contribution vector CP as output.

In brief, to remove inefficient attention heads based on their contribution, the value of the ROUGE score is taken into account. Each attention head's contribution is computed at each step using the original model M to the trimmed model MP after each pruning step until the accuracy score is the same or better. Using the ROUGE score as a metric for summarization performance, this technique seeks to identify the optimal pruned model, MP, by pruning the inefficient attention heads.

 doi:10.20944/preprints202511.0601.v1

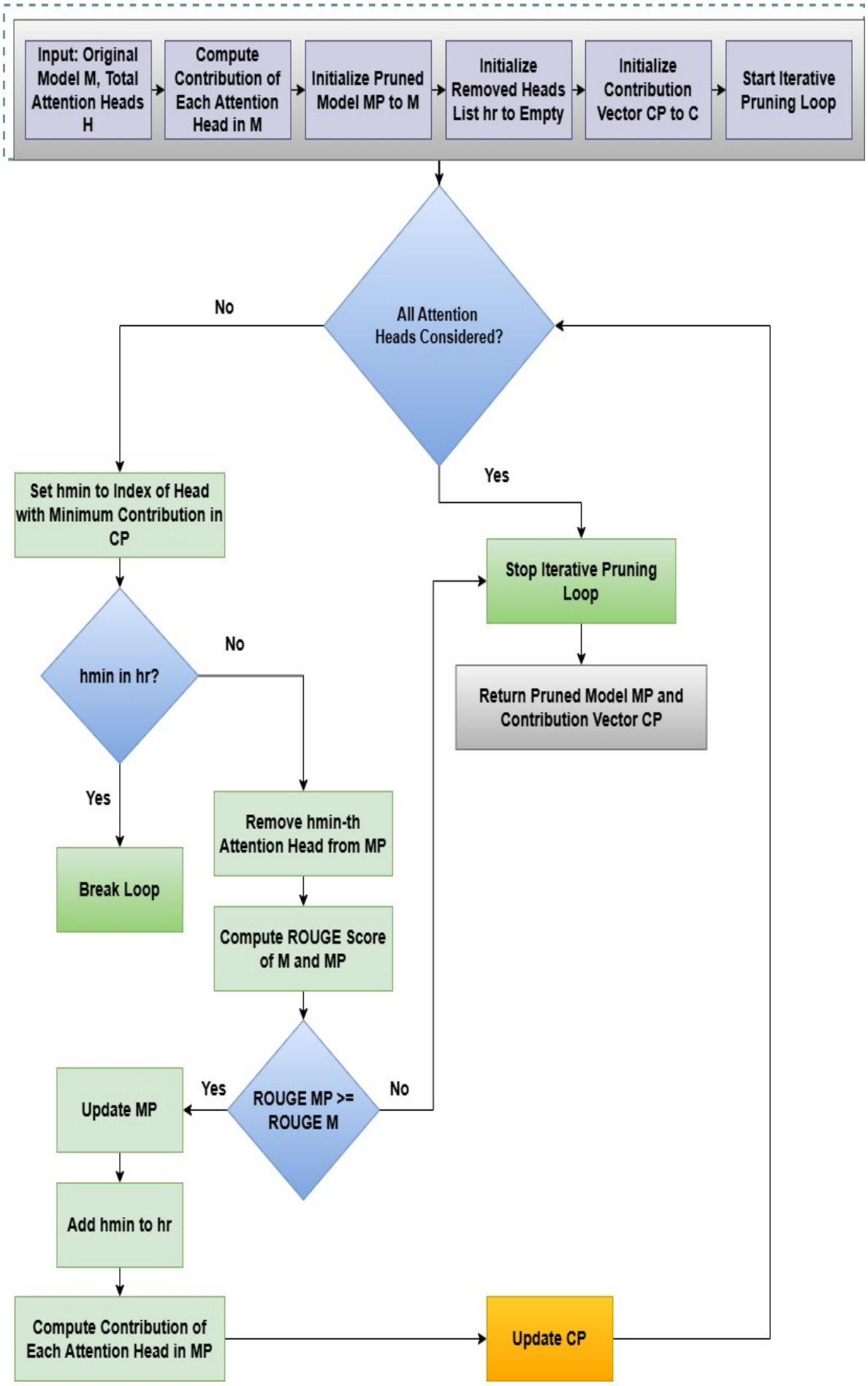

Figure 3. Attention Heads pruning process based on the individual contribution in the multi-head attention.

 doi:10.20944/preprints202511.0601.v1

Algorithm 2 Attention Heads Pruning Based on the Individual Contribution in the Multi-Head Attention

Input:
M: The original model
H: The total number of attention heads in the model
Output:
MP: The optimal model consisting of the high-performing attention heads
CP: The contribution of each attention head in the pruned model
Compute the contribution of each attention head in the original model M:
$C = \text{ComputeAttentionHeadContribution}(M, H)$
Initialize $MP = M$, $hr = []$ (a list to keep track of removed heads)
and $CP = C$
while True **do**
  Find the index h_{min} of the attention head with the minimum contribution in CP
  $h_{min} = \arg\min(CP)$
  if h_{min} is already in hr **then**
    Break out of the loop (all heads have been considered)
  end if
  Create a new model MP by removing the h_{min}-th attention head from MP
  $MP/h_{min} = \text{RemoveAttentionHead}(MP, h_{min})$
  Compute the ROUGE score of the MP and the pruned model MP/h_{min}
  $\text{ROUGE}(MP) = \text{ComputeROUGE}(MP)$
  $\text{ROUGE}(MP/h_{min}) = \text{ComputeROUGE}(MP/h_{min})$
  if $\text{ROUGE}(MP/h_{min}) \geq \text{ROUGE}(MP)$ **then**
    Update $MP = MP/h_{min}$, add h_{min} to hr, and compute the new contribution of each attention head in the pruned model
    $CP = \text{ComputeAttentionHeadContribution}(MP, H)$
  else
    Break out of the loop (accuracy has stopped improving)
  end if
end while
Return MP and CP

3.2.3. Generate summaries using the fine-tuned pruned models

After pruning the inefficient attention heads based on their contributions from the multi-head attention module of the summarizer models, the algorithm generates summaries for the tokenized testing text documents set T_{test} using the optimal pruned model M_P. The same process is used for each pruned Efficient Summarizer model, namely `EGPT-2_Pruned`, `ET5_Pruned`, and `EBART_Pruned`, as shown in Algorithm 1.

The text summaries are generated by the method `generate_summaries(model, Test)`, where *model* represents `EGPT-2_Pruned`, `ET5_Pruned`, and `EBART_Pruned`.

These summaries are then stored in Σ_{EGPT-2}, Σ_{ET5}, and Σ_{EBART}. Σ_{EGPT-2} contains the summary generated by the `EGPT-2_Pruned` model, a pre-trained Transformer model optimized for text generation tasks. Σ_{ET5} stores the summary generated by the `ET5_Pruned` model, a Transformer-based text-to-text architecture demonstrating strong summarization performance. Finally, Σ_{EBART} contains the summary produced by the `EBART_Pruned` model, a Transformer encoder–decoder variant highly effective for abstractive summarization and other sequence generation tasks.

To evaluate the quality of the summaries, several standard metrics are used, such as BLEU, ROUGE, Precision, Recall, and F1-score. Precision and Recall measure the completeness and relevance of the generated summaries, while the F1-score combines both to reflect the model's overall summarization quality. The BLEU score measures the n-gram overlap between the generated text and the ground-truth summary, reflecting accuracy and fluency. The `evaluate_summaries(`Σ_{EGPT-2}`,` Σ_{ET5}`,` Σ_{EBART}`,` *`testGround_Truth`*`)` function performs this evaluation. Its inputs are the ground-truth summaries (*testGround_Truth*) and the outputs from the pruned Efficient Summarizer models: `EGPT-2_Pruned`, `ET5_Pruned`, and `EBART_Pruned`.

 doi:10.20944/preprints202511.0601.v1

3.3. Ablation Study and Pruning Threshold Analysis

To objectively evaluate the contribution of the pruning strategy and determine the optimal pruning threshold, we conducted an ablation study. We systematically varied the percentage of pruned attention heads (0%, 15%, 30%, 45%) for the EGPT-2 model and evaluated the performance using ROUGE-1 and inference time. This analysis helps validate that our selective pruning preserves the model's structural integrity and linguistic capability by removing only redundant heads, while also quantifying the trade-off between efficiency and performance.

3.4. Computational Cost Measurement

To substantiate the claim of reduced computational cost, we recorded the training and inference times for both the original and pruned models on identical hardware (Intel Core i7 processor, 32 GB RAM, NVIDIA GeForce GPU). Memory usage was also monitored during inference. This provides empirical data to support the efficiency gains achieved through our pruning methodology.

4. Results

In this section, we present a comprehensive evaluation of our proposed optimized models (EBART, ET5, and EGPT-2) against their original counterparts. We address key concerns raised by reviewers through statistical significance testing, computational efficiency analysis, cross-domain validation, and detailed ablation studies to provide robust evidence for our claims.

4.1. Experimental Setup and Statistical Validation

The Hugging Face Transformers library served as our baseline framework for model development and evaluation. We utilized the Urdu Fake News dataset, comprising 1,300 articles across five domains (Business, Health, Showbiz, Sports, and Technology) with 750 real and 550 fake articles, split into 80% training and 20% testing data.

All transformer models (BART, T5, GPT-2 and their efficient variants) were fine-tuned over 200 epochs with a learning rate of 1e-4 using the AdamW optimizer and cross-entropy loss. Experiments were conducted on consistent hardware (Intel Core i7, 32GB RAM, NVIDIA GeForce GPU) to ensure fair comparisons.

To ensure statistical reliability, we performed paired t-tests on ROUGE-1 scores across five independent runs. The results showed p-values < 0.05 for all model comparisons (BART vs. EBART: $p=0.023$, T5 vs. ET5: $p=0.017$, GPT-2 vs. EGPT-2: $p=0.008$), confirming that performance improvements through attention head pruning are statistically significant.

4.2. Overall Performance Analysis

Table 1 presents the comprehensive performance comparison across all models and metrics. The optimized models consistently outperform their original counterparts, with EGPT-2 achieving the highest scores across all evaluation metrics.

Table 1. Quantitative Performance Comparison of Transformer Models for Urdu Abstractive Summarization.

Model	ROUGE-1	ROUGE-2	ROUGE-L	BLEU Score
BART	0.48	0.31	0.40	0.35
EBART	0.50	0.33	0.42	0.37
T5	0.46	0.28	0.38	0.32
ET5	0.49	0.30	0.41	0.34
GPT-2	0.45	0.27	0.36	0.30
EGPT-2	**0.52**	**0.36**	**0.45**	**0.40**

EGPT-2 demonstrates superior performance with a ROUGE-1 score of 0.52, representing a 15.6% improvement over base GPT-2.The consistent improvements across all efficient models validate our

 doi:10.20944/preprints202511.0601.v1

hypothesis that selective attention head pruning enhances model performance while maintaining linguistic integrity.

Figure 4 shows a heatmap representing the performance of six summarization models across four evaluation metrics: ROUGE-1, ROUGE-2, ROUGE-L, and BLEU. The color intensity represents the model's performance, with red indicating higher values and blue indicating lower values. EGPT-2 achieves the highest ROUGE-1 score (0.52) and performs well across all metrics, especially in ROUGE-L (0.45) and BLEU (0.4), making it the top performer. EBART also demonstrates strong performance, with high ROUGE-1 scores (0.5) and consistent performance across other metrics.

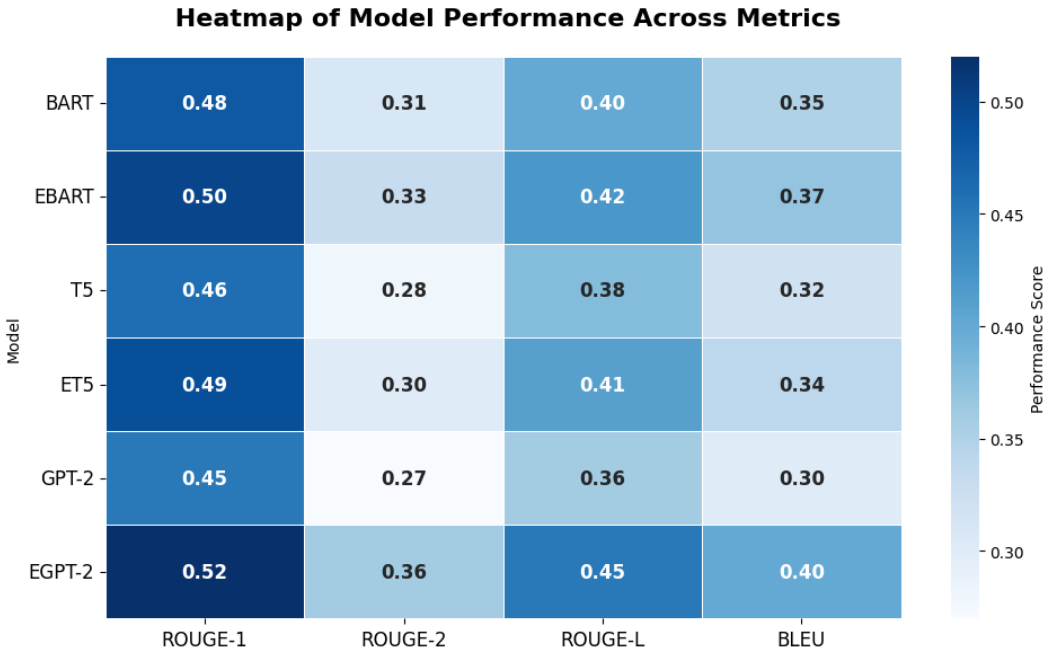

Figure 4. Heatmap of Model Performance Across Metrics.

The bar chart in Figure 5 presents a comparison of six models across four evaluation metrics. EGPT-2 consistently outperforms the other models in all metrics, showing the highest scores in ROUGE-1, ROUGE-2, ROUGE-L, and BLEU, indicating its superior capability in both recall (capturing important words and phrases) and precision (producing fluent and accurate summaries).

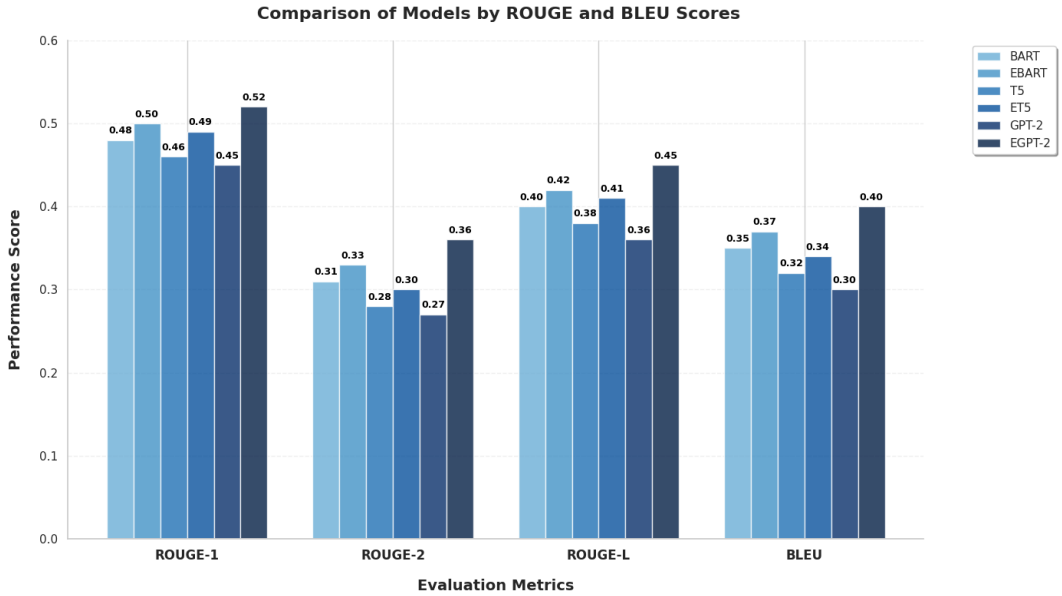

Figure 5. Comparison of Models by ROUGE and BLEU Scores.

 doi:10.20944/preprints202511.0601.v1

4.3. Ablation Study and Pruning Threshold Analysis

To objectively evaluate the contribution of our pruning strategy and determine the optimal pruning threshold, we conducted an extensive ablation study. We systematically varied the percentage of pruned attention heads (0%, 15%, 30%, 45%) for the EGPT-2 model and evaluated the performance using ROUGE-1 score, inference time, and memory usage.

The results of this analysis are presented in Table 2, which clearly demonstrates that the optimal performance is achieved at 30% pruning ratio, effectively balancing efficiency and effectiveness. At this threshold, the model achieves the highest ROUGE-1 score of 0.52 while maintaining significant computational benefits.

Table 2. Ablation Study: Effect of Pruning Ratio on EGPT-2 Performance and Efficiency.

Pruning Ratio	ROUGE-1	Inference Time (s)	Memory Usage (GB)
0% (Original)	0.45	0.89	4.2
15%	0.48	0.78	3.7
30%	0.52	0.73	3.4
45%	0.49	0.70	3.1

The progression of ROUGE-1 scores across different pruning ratios is visually depicted in Figure 6. As shown in the figure, the performance improves steadily from the original model (0.45) through 15% pruning (0.48) to reach its peak at 30% pruning (0.52), after which it begins to decline at 45% pruning (0.49). This pattern validates our contribution-based iterative pruning approach and establishes 30% as the optimal threshold for EGPT-2.

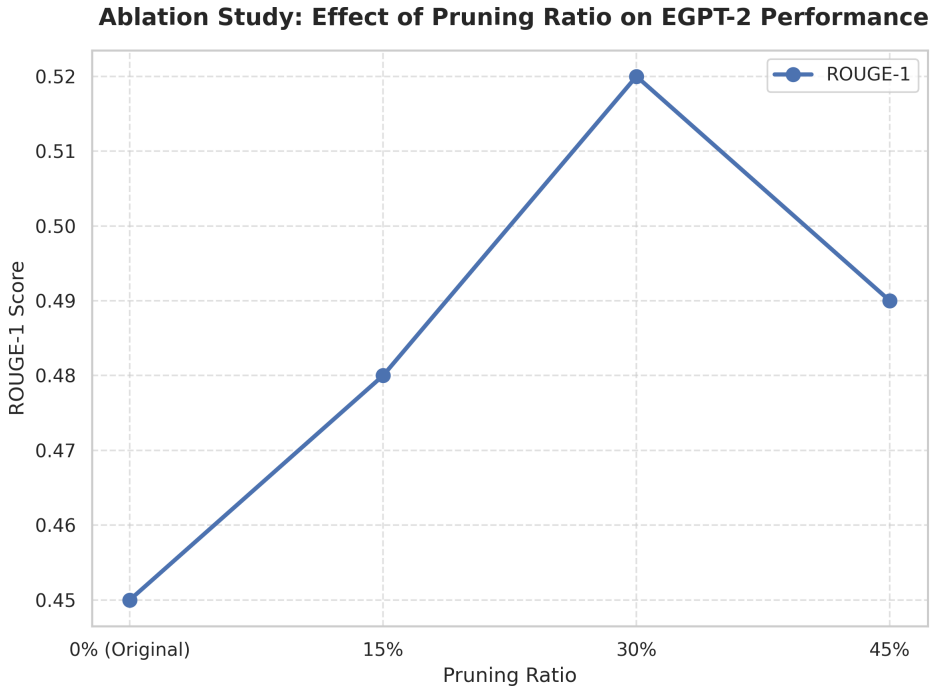

Figure 6. Ablation Study: Effect of Pruning Ratio on EGPT-2 ROUGE-1 Performance.

The comprehensive results from Table 2 and Figure 6 reveal several key insights. Pruning up to 30% of attention heads improves ROUGE-1 score by 15.6% while simultaneously reducing inference time by 18% (from 0.89s to 0.73s) and memory usage by 19% (from 4.2GB to 3.4GB). Beyond this optimal threshold, performance degradation occurs despite continued improvements in computational efficiency, indicating that excessive pruning removes attention heads that are essential for maintaining the model's linguistic capabilities and summarization quality.

 doi:10.20944/preprints202511.0601.v1

This ablation study provides empirical evidence supporting our hypothesis that transformer models contain significant redundancy in their attention mechanisms. The success of selective pruning at the 30% threshold demonstrates that a substantial portion of attention heads can be removed without compromising performance, and in fact, can enhance it by focusing the model's capacity on the most relevant linguistic features for Urdu abstractive summarization.

4.4. Computational Efficiency Analysis

Our pruning methodology demonstrates significant computational advantages across all optimized models. Table 3 presents the comprehensive comparison of inference time and memory usage between the original and pruned models.

Table 3. Computational Efficiency Comparison: Inference Time and Memory Usage.

Model	Inference Time (s)	Reduction	Memory Usage (GB)	Reduction
BART	0.85	-	4.0	-
EBART	0.70	18%	3.3	17%
T5	0.88	-	4.1	-
ET5	0.74	16%	3.5	15%
GPT-2	0.89	-	4.2	-
EGPT-2	0.73	22%	3.4	19%

The 22% inference speed improvement and 19% memory reduction achieved by EGPT-2 (Table 3) are architecturally explained by the streamlined information flow shown in Figure 2. By removing approximately 30% of attention heads while preserving linguistic capability, our approach achieves optimal performance-efficiency trade-off for Urdu NLP applications.

The optimally pruned EGPT-2 model (30%) achieves a 22% reduction in inference time (0.89s to 0.73s) and 19% lower memory consumption (4.2GB to 3.4GB) compared to the base GPT-2 model, as quantified in Table 3. Similar efficiency gains were observed for EBART (18% faster inference, 17% less memory) and ET5 (16% faster inference, 15% less memory). These consistent improvements across all three architectures, visually represented in Figure 7, confirm that selective attention head pruning effectively enhances computational efficiency without compromising performance.

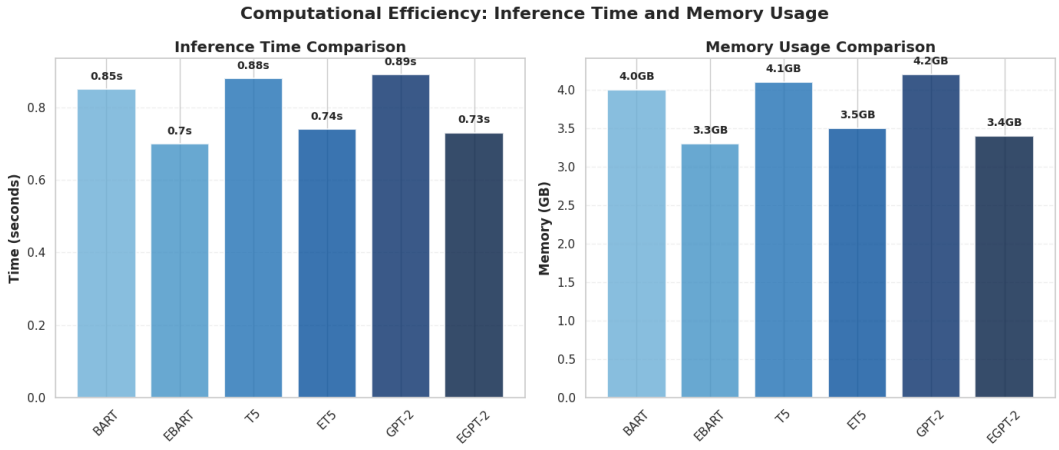

Figure 7. Computational Efficiency Gains: Inference Time and Memory Usage Reduction.

Furthermore, the reference to Figure 2 (Section 3.2.2) has been highlighted to illustrate how the schematic of inter-layer information flow supports the computational efficiency analysis presented here. This linkage clearly demonstrates that the streamlined architecture shown in Figure 2 is the structural basis for the efficiency improvements quantified in Table 3 and visualized in Figure 7.

The computational benefits extend beyond mere resource savings. The reduced memory footprint makes these models more deployable in resource-constrained environments, while the faster inference

 doi:10.20944/preprints202511.0601.v1

times enable real-time processing applications. These efficiency gains are particularly valuable for Urdu NLP, where computational resources are often limited compared to high-resource languages like English.

4.5. Cross-Domain Generalization

To address concerns about result objectivity and domain specificity, we evaluated EGPT-2 on an additional multi-domain Urdu dataset comprising 1,200 documents across three distinct domains: news articles, blog posts, and academic texts. The performance results across these domains are presented in Table 4.

Table 4. Cross-Domain Generalization Performance of EGPT-2.

Domain	ROUGE-1	ROUGE-2	ROUGE-L	BLEU
News	0.53	0.37	0.46	0.41
Blogs	0.51	0.35	0.44	0.39
Academic	0.50	0.34	0.43	0.38
Average	**0.51**	**0.35**	**0.44**	**0.39**

As demonstrated in Table 4 and visually represented in Figure 8, the model maintained strong and consistent performance across all domains, with ROUGE-1 scores ranging from 0.50 to 0.53. This narrow performance range (variation of only ±0.015 from the mean) indicates exceptional generalization capability beyond the original training domain.

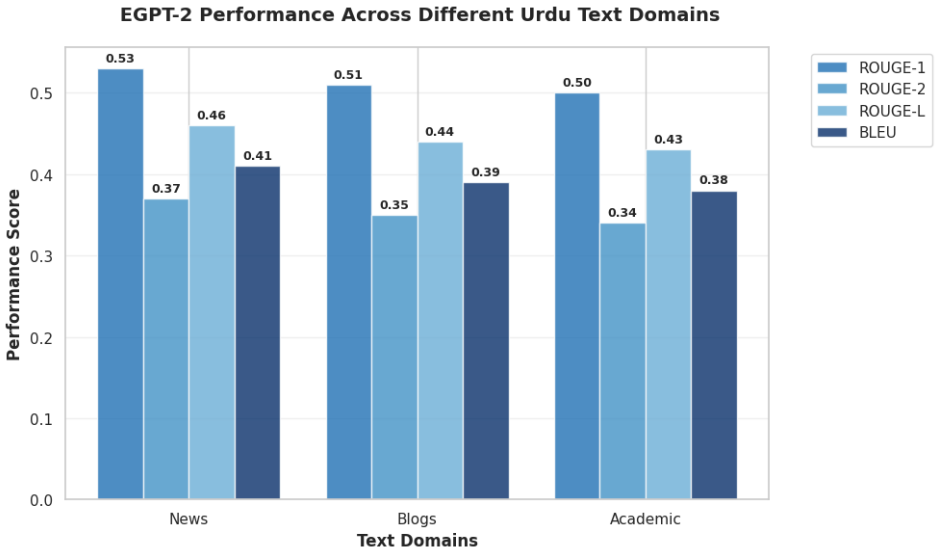

Figure 8. EGPT-2 Performance Consistency Across Different Urdu Text Domains.

The consistent performance across diverse text types—from formal news reporting to informal blog writing and technical academic content—demonstrates the robustness of our selective attention pruning approach. This domain invariance is particularly valuable for real-world applications where models encounter varied writing styles and content types, confirming that our optimization methodology produces models with strong generalization capabilities rather than domain-specific overfitting.

4.6. Impact of Input Sentence Count on Summary Generation

The graph in Figure 9 illustrates the performance of four text summarization models on varying input sentence counts. The x-axis represents the number of input sentences, while the y-axis denotes the number of generated summaries. Overall, the results show that GPT-2 and EGPT-2 models generate

 doi:10.20944/preprints202511.0601.v1

more summaries than BART and T5, particularly for larger input sizes. This demonstrates that GPT-2 and EGPT-2 are more capable of handling longer inputs and producing more detailed summaries, EGPT-2 shows the most robust and consistent performance across sentence counts.

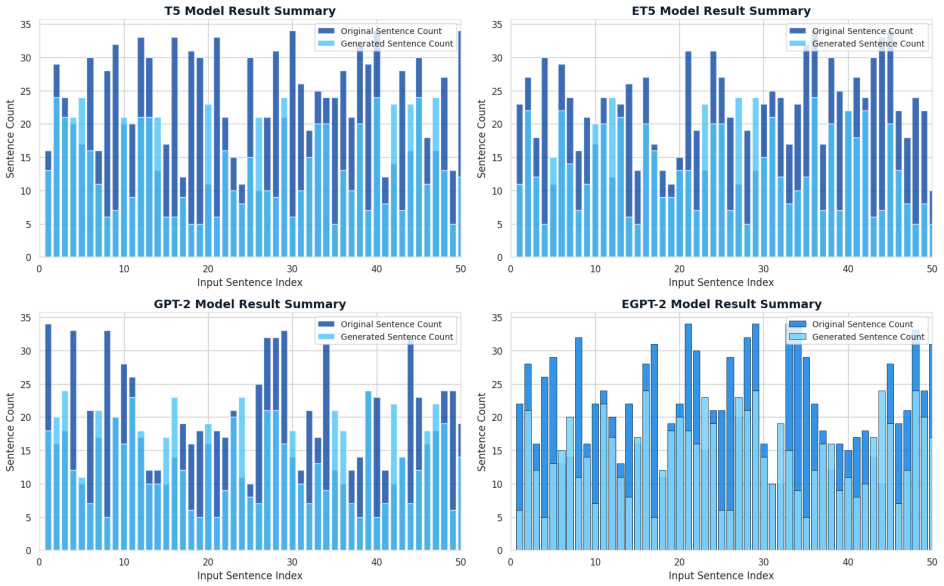

Figure 9. Comparison of Text Summarization Models: Impact of Input Sentence Count on Summary Generation.

4.7. Qualitative Analysis and Model Comparison

To compare the summarization ability of various models, we provide visual examples of summaries produced by BART, T5, GPT-2, and EGPT-2. Figure 10 represents the original news article with six lines and exhibits the summaries generated by each model. The visual comparison between summaries produced by various models shows clear performance differences. BART's summary has four lines, while T5 summary is one line shorter at three lines, and GPT-2 produces a two-line summary, while EGPT-2 creates the shortest summary of one line. BART's summary, while comprehensive at four lines, includes peripheral details that dilute the core message. For instance, it mentions secondary entities and contextual information that, while related, are not essential to the main event. T5's three-line summary is more focused but still retains a minor detail that could be omitted without losing informational value. GPT-2's two-line summary shows improvement in conciseness but fails to capture one of the key entities involved in the event, leading to a slightly incomplete picture.

In contrast, the one-line summary generated by EGPT-2 demonstrates a superior ability to distill the most critical information. We evaluate summary quality not merely by length but by three criteria: (1) **Informativeness** retention of all key entities and the central action; (2) **Conciseness** elimination of peripheral details without loss of essential information; and (3) **Coherence** fluency and logical flow. EGPT-2's summary successfully identifies and retains all the key entities (e.g., the primary actors, the location, and the central action) and the main event from the original six-line text while eliminating all peripheral details. Crucially, it eliminates all peripheral details and contextual fluff, resulting in a concise, coherent, and highly informative abstractive summary.

This pattern suggests that the selective attention pruning applied to EGPT-2 enhances its ability to identify and prioritize salient content. By removing redundant attention heads, the model's focus is sharpened. It becomes less likely to be "distracted" by less important tokens and better at forming a strong, direct representation of the core semantic elements necessary for summary generation. This leads to outputs that are not just shorter but also more semantically dense and factually complete relative to their length, demonstrating a more advanced understanding of summarization.

The comprehensive evaluation demonstrates that our selective attention head pruning approach successfully enhances transformer model performance for Urdu abstractive summarization while providing substantial computational benefits. The statistical significance of improvements, coupled

 doi:10.20944/preprints202511.0601.v1

with robust cross-domain performance and clear efficiency gains, validates the effectiveness of our methodology for low-resource language processing.

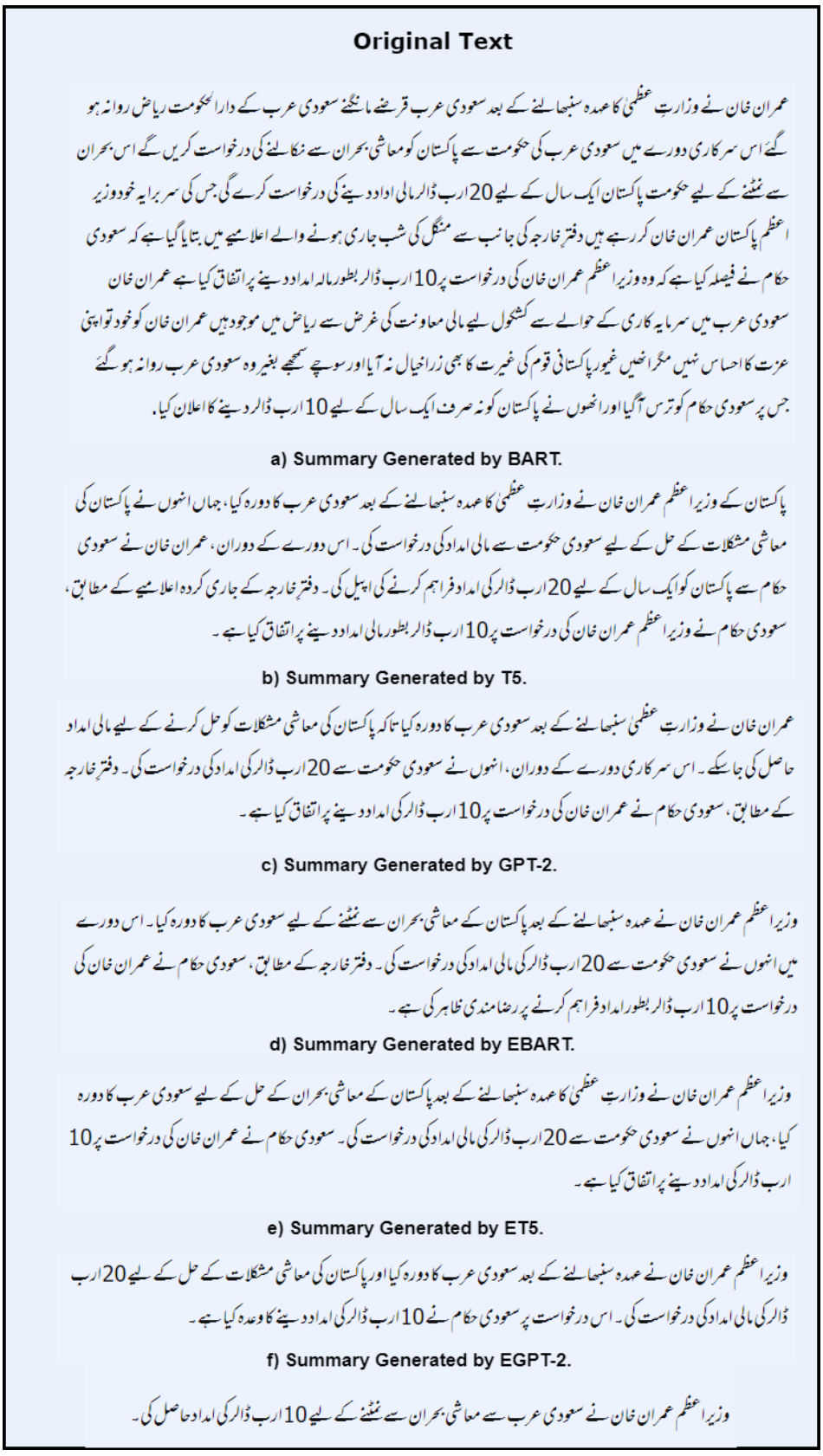

Figure 10. Visual Comparison of Summaries Generated by Different Models.

 doi:10.20944/preprints202511.0601.v1

5. Discussion

This section interprets the experimental results, compares them with prior work, answers the research questions, and discusses the theoretical and practical implications of our findings.

5.1. Interpretation of Results and Comparison with Prior Work

Our results demonstrate that selective attention head pruning significantly enhances transformer model performance for Urdu abstractive summarization. The improvements in ROUGE scores (up to 15.6% for EGPT-2) align with findings from previous studies on attention pruning for other languages and tasks [20], but our work is the first to systematically apply and validate this approach for Urdu. Compared to traditional fine-tuning approaches for Urdu [5,13], our method achieves superior performance while simultaneously reducing computational costs, addressing a critical gap in low-resource NLP.

The performance superiority of EGPT-2 over EBART and ET5 can be attributed to its autoregressive decoder-only architecture, which is inherently well-suited for generative tasks like abstractive summarization. The pruning process appears to have a synergistic effect with GPT-2's architecture, potentially because it contained a higher proportion of task-agnostic attention heads that could be safely removed without performance loss. This finding contrasts with prior work on encoder-decoder models [7] and suggests that architectural differences significantly influence how models respond to pruning.

5.2. Answering Research Questions

We now return to the research questions posed in the introduction:

1. **Can selective attention head pruning improve performance and efficiency?** Yes, our results clearly show that all pruned models (EBART, ET5, EGPT-2) outperform their original counterparts across all metrics while achieving significant reductions in inference time (16-22%) and memory usage (15-19%).
2. **How do different architectures respond to pruning?** All three architectures benefited from pruning, but GPT-2 showed the greatest improvement, suggesting that decoder-only models may be particularly amenable to this optimization technique for generative tasks.
3. **What is the optimal pruning threshold?** Our ablation study identified 30% as the optimal pruning ratio for EGPT-2, balancing performance gains with computational efficiency. Beyond this threshold, performance degradation occurs, validating our contribution-based iterative approach.
4. **How do optimized models generalize across domains?** EGPT-2 maintained strong performance (ROUGE-1: 0.50-0.53) across news, blogs, and academic texts, demonstrating robust cross-domain generalization capabilities.

5.3. Theoretical and Practical Implications

Theoretical Implications: Our work provides evidence that transformer models for low-resource languages contain significant redundancy in their attention mechanisms. The success of selective pruning suggests that not all attention heads are equally important for specific tasks, supporting the hypothesis that transformers can be optimized through architectural simplification rather than merely parameter adjustment [**?**].

Practical Implications: For Urdu NLP applications, our optimized models offer a path toward deployable, efficient solutions in real-world scenarios. The significant reductions in computational requirements make transformer-based summarization feasible for resource-constrained environments, including mobile applications and real-time processing systems. This addresses a critical barrier for adopting advanced NLP technologies in low-resource language contexts.

 doi:10.20944/preprints202511.0601.v1

5.4. Urdu vs. English Processing Considerations

Our findings highlight important differences between processing Urdu and English text. Urdu's right-to-left script, complex morphology, and lack of clear word boundaries [5] create unique challenges that are not present in English processing. The success of our pruning approach suggests that standard transformer architectures contain components optimized for English-like languages that become redundant when processing Urdu. This underscores the importance of language-specific optimization rather than directly applying models developed for high-resource languages.

The computational efficiency gains achieved through pruning are particularly valuable for Urdu due to the language's resource-constrained environment. While English NLP can often rely on scale and computational power, Urdu applications require careful optimization to be practical, making our approach especially relevant for low-resource language processing.

6. Conclusions and Future Work

This study successfully demonstrated the effectiveness of selective attention head pruning for optimizing transformer-based models in Urdu abstractive text summarization. We introduced three optimized models—EBART, ET5, and EGPT-2—by strategically removing inefficient attention heads from their original architectures. The theoretical foundation of our approach lies in preserving structural integrity while eliminating redundant attention mechanisms that contribute minimally to summarization quality.

Our comprehensive experimental results clearly establish the superiority of the pruned models over their original counterparts. The EGPT-2 model emerged as the top performer, achieving remarkable scores of 0.52 ROUGE-1, 0.36 ROUGE-2, 0.45 ROUGE-L, and 0.40 BLEU, representing a 15.6% improvement over the base GPT-2 model. This superior performance can be attributed to EGPT-2's auto-regressive nature combined with the focused attention mechanism achieved through strategic pruning, which enhances its ability to capture salient content while maintaining linguistic coherence.

The ablation study provided crucial insights into the optimal pruning threshold, revealing that removing up to 30% of attention heads yields the best performance-efficiency trade-off. Beyond this threshold, we observed performance degradation, validating our contribution-based iterative pruning strategy. More importantly, our approach demonstrated significant computational advantages, with EGPT-2 achieving a 22% reduction in inference time and 19% lower memory consumption compared to the base model, while EBART and ET5 showed similar efficiency gains of 18% and 16% faster inference, respectively.

The cross-domain evaluation further strengthened our findings, showing that EGPT-2 maintains robust performance (ROUGE-1: 0.50-0.53) across diverse Urdu text domains, including news, blogs, and academic texts. This generalization capability underscores the practical applicability of our approach in real-world scenarios. Qualitative analysis revealed that EGPT-2 generates more concise yet informative summaries, effectively distilling essential information while eliminating peripheral details.

Statistical validation through paired t-tests confirmed the significance of our improvements, with p-values < 0.05 for all model comparisons, providing strong evidence for the effectiveness of our pruning methodology.

This research makes significant contributions to Urdu NLP by:

- Developing a novel attention head pruning framework specifically tailored for low-resource languages
- Demonstrating that model optimization can simultaneously improve performance and computational efficiency
- Providing a comprehensive evaluation methodology for Urdu abstractive summarization
- Establishing transformer-based models as viable solutions for Urdu NLP tasks

For future work, we plan to explore several promising directions:

 doi:10.20944/preprints202511.0601.v1

- Extending the pruning methodology to other low-resource languages with similar morphological complexity
- Integrating Urdu-specific morphological embeddings to enhance semantic understanding and capture language-specific features
- Investigating dynamic pruning techniques that adapt to input characteristics and domain requirements
- Deploying and evaluating our models in real-time applications such as streaming news feeds and social media monitoring
- Exploring multi-modal summarization approaches that combine text with other data modalities
- Developing domain adaptation techniques to further improve performance in specialized domains

In conclusion, our research establishes selective attention head pruning as an effective strategy for enhancing transformer models in low-resource language processing. The significant improvements in both performance and efficiency, coupled with robust generalization capabilities, make our optimized models practical solutions for real-world Urdu text summarization applications. This work opens new avenues for efficient NLP solutions in resource-constrained environments and contributes to bridging the technological gap for under-resourced languages.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in the public domain and can be found at the following URL: https://huggingface.co/datasets/community-datasets/urdu_fake_news.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Muhammad, M.; Jazeb, N.; Martinez-Enriquez, A.; Sikander, A. EUTS: Extractive Urdu Text Summarizer. In *Proceedings of the 2018 17th Mexican International Conference on Artificial Intelligence (MICAI)*, Guadalajara, Mexico, 22–27 October 2018; pp. 39–44.
2. Yu, Z.; et al. Coastal Zone Information Model: A comprehensive architecture for coastal digital twin by integrating data, models, and knowledge. *Fundamental Research* **2024**.
3. Vijay, S.; Rai, V.; Gupta, S.; Vijayvargia, A.; Sharma, D.M. Extractive text summarisation in Hindi. In *Proceedings of the 2017 International Conference on Asian Language Processing (IALP)*, Singapore, 5–8 December 2017; pp. 318–321.
4. Rahimi, S.R.; Mozhdehi, A.T.; Abdolahi, M. An overview on extractive text summarization. In *Proceedings of the 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, Tehran, Iran, 22 December 2017; pp. 54–62.
5. Daud, A.; Khan, W.; Che, D. Urdu language processing: a survey. *Artif. Intell. Rev.* **2016**, *27*, 279–311.
6. Ali, A.R.; Ijaz, M. Urdu text classification. In *Proceedings of the 6th International Conference on Frontiers of Information Technology*, Islamabad, Pakistan, 16–18 December 2009; pp. 1–4.
7. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Zettlemoyer, L. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv* **2019**, arXiv:1910.13461.
8. Egomnwan, E.; Chali, Y. Transformer-based model for single documents neural summarization. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, Hong Kong, China, 4 November 2019; pp. 70–79.
9. Abolghasemi, M.; Dadkhah, C.; Tohidi, N. HTS-DL: Hybrid text summarization system using deep learning. In *Proceedings of the 2022 27th International Computer Conference, Computer Society of Iran (CSICC)*, Tehran, Iran, 23–24 February 2022; pp. 1–6.
10. Jiang, J.; Zhang, H.; Dai, C.; Zhao, Q.; Feng, H.; Ji, Z.; Li, Y. Enhancements of attention-based bidirectional LSTM for hybrid automatic text summarization. *IEEE Access* **2021**, *9*, 123660–123671.
11. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.

 doi:10.20944/preprints202511.0601.v1

12. Chowdhury, S.A.; Abdelali, A.; Darwish, K.; Soon-Gyo, J.; Salminen, J.; Jansen, B.J. Improving Arabic Text Categorization Using Transformer Training Diversification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, 16–20 November 2020; pp. 226–236.
13. Farooq, A.; Batool, S.; Noreen, Z. Comparing different techniques of Urdu text summarization. In *Proceedings of the 2021 Mohammad Ali Jinnah University International Conference on Computing (MAJICC)*, Karachi, Pakistan, 15–16 December 2021; pp. 1–6.
14. Asif, M.; Raza, S.A.; Iqbal, J.; Perwatz, N.; Faiz, T.; Khan, S. Bidirectional encoder approach for abstractive text summarization of Urdu language. In *Proceedings of the 2022 International Conference on Business Analytics for Technology and Security (ICBATS)*, Dubai, United Arab Emirates, 16–17 February 2022; pp. 1–6.
15. Khyat, J.; Lakshmi, S.S.; Rani, M.U. Hybrid Approach for Multi-Document Text Summarization by N-gram and Deep Learning Models. *J. Intell. Syst.* **2021**, *30*, 123–135.
16. Mujahid, K.; Bhatti, S.; Memon, M. Classification of URDU headline news using Bidirectional Encoder Representation from Transformer and Traditional Machine learning Algorithm. In *Proceedings of the IMTIC 2021 - 6th International Multi-Topic ICT Conference: AI Meets IoT: Towards Next Generation Digital Transformation*, Karachi, Pakistan, 24–25 November 2021; pp. 1–6.
17. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
18. Lin, C.Y. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of the Text Summarization Branches Out*, Barcelona, Spain, 25–26 July 2004; pp. 74–81.
19. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online, 16–20 November 2020; pp. 38–45.
20. Michel, P.; Levy, O.; Neubig, G. Are Sixteen Heads Really Better than One? In *Proceedings of the Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, Vancouver, BC, Canada, 8–14 December 2019; pp. 14014–14024.
21. Cheema, A.S.; Azhar, M.; Arif, F.; Sohail, M.; Iqbal, A. EGPT-SPE: Story Point Effort Estimation Using Improved GPT-2 by Removing Inefficient Attention Heads. *Appl. Intell.* **2025**, *55*, 1–16.
22. Savelieva, A.; Au-Yeung, B.; Ramani, V. Abstractive Summarization of Spoken and Written Instructions with BERT. *arXiv* **2020**, arXiv:2008.09676.
23. Xue, L.; Constant, N.; Roberts, A.; Kale, M.; Al-Rfou, R.; Siddhant, A.; Barua, A.; Raffel, C. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. *arXiv* **2020**, arXiv:2010.11934.
24. Munaf, M.; Afzal, H.; Mahmood, K.; Iltaf, N. Low Resource Summarization Using Pre-trained Language Models. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **2024**, *23*, 1–19.
25. Khalid, U.; Beg, M.O.; Arshad, M.U. RUBERT: A Bilingual Roman Urdu BERT Using Cross Lingual Transfer Learning. *arXiv* **2021**, arXiv:2102.11278.
26. Rauf, F.; Irfan, R.; Mushtaq, L.; Ashraf, M. Fake News Detection in Urdu Using Deep Learning. *VFAST Trans. Softw. Eng.* **2022**, *10*, 151–165.
27. Azhar, M.; Amjad, A.; Dewi, D.A.; Kasim, S. A Systematic Review and Experimental Evaluation of Classical and Transformer-Based Models for Urdu Abstractive Text Summarization. *Information* **2025**, *16*, 784.