ResFormer: All-Time Reservoir Memory for Long Sequence Classification

Hongbo Liu* **Jia Xu** *
Gateway Academic Center, Stevens Institute of Technology
601 Hudson St, Hoboken, NJ 07030
hliu93@stevens.edu, jxu70@stevens.edu

Abstract

Sequence classification is essential in NLP for understanding and categorizing language patterns in tasks like sentiment analysis, intent detection, and topic classification. Transformer-based models, despite achieving state-of-the-art performance, have inherent limitations due to quadratic time and memory complexity, restricting their input length. Although extensive efforts have aimed at reducing computational demands, processing extensive contexts remains challenging.

To overcome these limitations, we propose ResFormer, a novel neural network architecture designed to model varying context lengths efficiently through a cascaded methodology. ResFormer integrates an reservoir computing network featuring a nonlinear readout to effectively capture long-term contextual dependencies in linear time. Concurrently, short-term dependencies within sentences are modeled using a conventional Transformer architecture with fixed-length inputs.

Experiments demonstrate that ResFormer significantly outperforms baseline models of DeepSeek-Qwen and ModernBERT, delivering an accuracy improvement of up to +22.3% on the EmoryNLP dataset and consistent gains on MultiWOZ, MELD, and IEMOCAP. In addition, ResFormer exhibits reduced memory consumption, underscoring its effectiveness and efficiency in modeling extensive contextual information.

1 Introduction

Transformer models have significantly advanced state-of-the-art performance across various natural language processing (NLP) tasks (Vaswani et al., 2017; Devlin et al., 2018; Dosovitskiy et al., 2020). However, an important limitation of Transformer architectures is their inherent quadratic complexity concerning input length, which restricts the number of tokens they can effectively process. For instance, popular models such as LlaMA 3 (Meta, 2024), Gemma (Team et al., 2024), GPT-4 (Achiam et al., 2023), and Mistral (Jiang et al., 2023) typically have a maximum input length limited to approximately $8K$ tokens.

Despite this limitation, tasks such as emotion detection greatly benefit from analyzing extensive sequential contexts to capture subtle emotional cues and evolving relational dynamics effectively. For example, in the TV series *Friends*, accurately understanding Joey's emotional development, from carefree bachelorhood to genuinely caring about his friends' happiness, necessitates context derived from numerous episodes spanning several seasons; Similarly, Monica's compulsive cleanliness, initially played for laughs, but across seasons, it reflects deep insecurity rooted in childhood neglect. Without long-term context, such emotional complexity is easily misclassified. Thus, efficiently modeling all context of the narrative is essential to adequately analyze emotion.

Many studies have investigated how to efficiently increase Transformer input lengths, such as (Munkhdalai et al., 2024; Tworkowski et al., 2024; Bertsch et al., 2024; Mohtashami and Jaggi, 2024). Existing solutions, however, either modifying the attention model with heuristic assumptions or projecting long input into a fixed dimension. Since most work does not consider temporal patterns of the input, there is still room to reduce the information loss learned from the long context and improve the prediction accuracy and efficiency.

In this work, we introduce a novel neural network architecture ResFormer that integrates Reservoir Computing Network and Transformer to efficiently handle long sequences. Echo State Network (Reservoir) is a kind of Reservoir Computing (RC) that is a class of simple and efficient Recurrent Neural Networks where internal weights are fixed at random, and only a linear output layer is trained.

*Authors are listed in alphabetical order.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 11244–11256
November 4-9, 2025

Reservoir requires a small number of training data samples and computing resources with great advantages for processing sequential data in linear time and constant space (Gauthier et al., 2021). Here, we improve Reservoir with nonlinear readout to take long conversational context into account for time and memory efficiency (Gauthier et al., 2021).

Notebly, our method is model agnostic, that means, our ResFormer can be also applied on any other neural network architecture such as convolutional neural network and LSTM to improve their performance.

ResFormer Architecture The core innovation of our ResFormer lies in the integration of Reservoir and Transformer models. Figure 1 depicts the architecture of our ResFormer, which consists of two cascaded modules tailored for different context lengths.

For longer contexts, an Reservoir models the entire context relevant to the current utterance, drawing from all available training corpora. For example, in a script containing multiple seasons and episodes, the Reservoir learns the emotion of a character based on the current sentence within the larger context throughout the TV series, i.e., scripts of all seasons and episodes. In scenarios where the corpus consists of several unrelated books, each book is learned in its entirety, with the Reservoir re-initialized randomly at the beginning of each book's first sentence.

For shorter contexts, a Transformer-based module captures token-level dependencies within individual sentences. Each sentence is initially processed by the Transformer-based model to encode within-sentence dependencies, generating sentence embeddings subsequently combined with outputs from the Reservoir module. The Reservoir reads sentence after sentence in their embedded forms and then update the Reservoir states on all the context of a corpus without training.

This combined representation is fed into a neural network block. The neural network block can be of any architecture, and in this paper, we showcase on the Transformer model. We apply the CLS token's hidden state to incorporate the sentence level information, enabling comprehensive contextual dependency modeling. This cascaded approach effectively models both long-term dependencies and conventional token-level relationships.

Our proposed reservoir method enables the Transformer architecture to process an unlimited number of input tokens. Our primary contributions are as follows:

1. We propose the ResFormer architecture, capable of theoretically handling arbitrarily long inputs with linear time complexity.

2. We develop a cascaded learning framework to effectively manage long and short contextual information separately.

3. We introduce a cross attention readout mechanism leveraging cross-attention, significantly enhancing the performance of Reservoir.

4. We empirically validate our method, demonstrating notable performance improvements on emotion and intent detection tasks.

Our experiments demonstrate that our ResFormer significantly enhances performance on NLP classification tasks. Specifically, we observe accuracy increase of +19.9% over ModernBERT (Warner et al., 2024) and +22.3% over DeepSeek-Qwen-1.5B (DeepSeek-AI et al., 2025) on the EmoryNLP dataset, and up to +8.58%, +8.00%, and +14.6% over ModernBERT (Warner et al., 2024), DeepSeek (DeepSeek-AI et al., 2025), and Longformer (Beltagy et al., 2020) on the MELD (Poria et al., 2019) dataset, respectively, in addition to the consistent improvements on MultiWOZ 2.2 (Zang et al., 2020) and IEMOCAP (Busso et al., 2008).

2 Problem Definition

A corpus, such as a book, a series of conversations, and all episodes and seasons of a TV-series, can be represented in the text form of a sequence of sentences $\mathbf{u}_1^I = \mathbf{u}_1, \mathbf{u}_2 \cdots \mathbf{u}_i \cdots \mathbf{u}_I$ $(i \in 1, 2, \cdots, I)$, here I is the total number of sentences in the corpus and each individual sentence $\mathbf{u_i}$ is defined as $\mathbf{u_i} = w_1 \cdots w_j \cdots w_J$, where J is the sentence length. A training dataset can be composed of one or multiple corpora.

In a textual sequence classification task, such as emotion and intent detection, we aim to predict the best class $\hat{c}$ for the sentence $\mathbf{u_i}$ based on all previous relevant sentences from $\mathbf{u_1^{i-1}}$:

$$\hat{c_i} = \underset{c}{\operatorname{argmax}} \Pr(c|\mathbf{u}_i; \mathbf{u}_1^{i-1}) \quad (1)$$

In Section 3, we will describe our ResFormer to infer $\Pr$ and predict the sequence class.

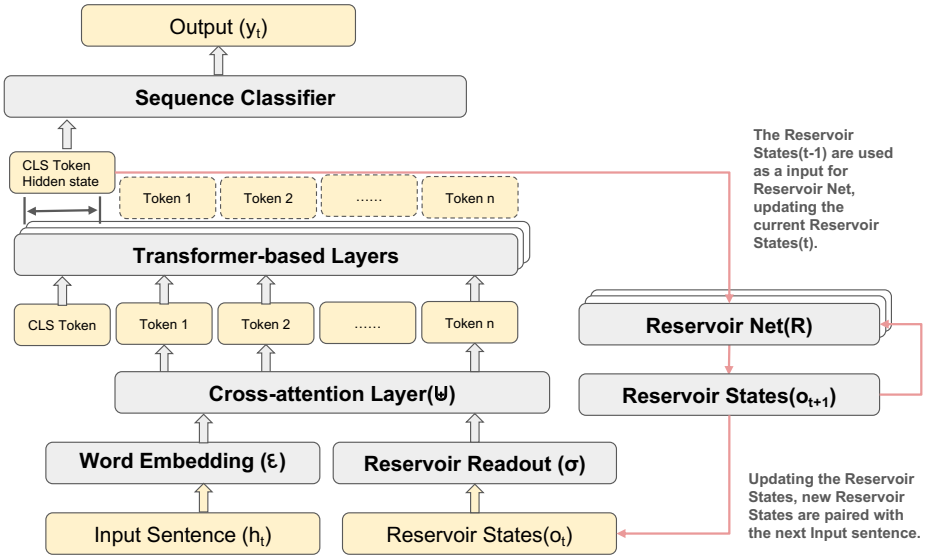

Figure 1: A schematic of ResFormer: input sentences are first processed by a word embedding layer, while Reservoir states are passed into a nonlinear readout layer. The resulting Reservoir outputs and input embeddings are then combined via a cross-attention layer and fed into the Transformer.

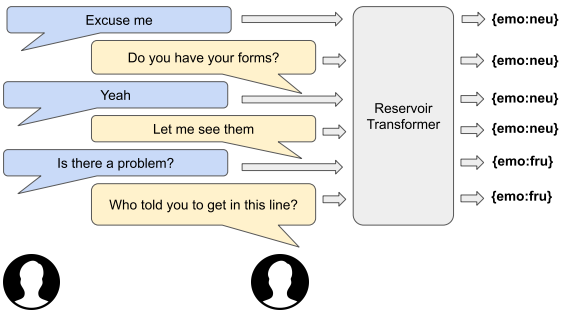

Figure 2: An example from the IEMOCAP dataset. Arrows on the left show how the Reservoir states are updated and feed into the transformer-based models as inputs.

3 ResFormer

Architecture ResFormer $\mathbb{RT}$ is named after reservoir computing (Gallicchio et al., 2017) $\mathbb{R}(\cdot)$ and Transformer $\mathbb{T}(\cdot)$ architecture it builds upon. The key innovation of $\mathbb{RT}$ lies in integrating two distinct memory mechanisms to capture information across different levels of context. This all-context-aware memory framework does not only tackles the challenge of processing long contexts but also selectively emphasizes the most relevant recent inputs. The two memory modules work in tandem to model dependencies at multiple scales when the context length grows:

- The long-term memory module (LTM) in Section 3.2 processes the entire context, allowing it to capture extended dependencies across all previous input sentences in a corpus.
- The short-term memory module (STM) in Section 3.1, on the other hand, focuses on local dependencies within individual sentences.

Together, these modules enable the model to efficiently manage and utilize context across both global and local levels. Now, we will describe these two modules to handle different ranges of input lengths, i.e., LTM realized with reservoir, while STM implemented as Transformer:

$$\hat{y}_i = \mathbb{T}(\mathbb{R}(\epsilon(\mathbf{u}_1^{i-1})) \uplus \epsilon(\mathbf{u}_i)) \tag{2}$$

Here, $\hat{y}$ is the final output of the class prediction $\hat{c}$ in Equation 1. The reservoir $\mathbb{R}(\cdot)$ takes all time inputs sequentially, i.e., the long-term historical context from the initial time step to the latest time step and outputs reservoir states as a compressed full context representation. The operator $\uplus$ represents a cross attention that combines input embeddings with reservoir state output, and then feeds it into a Transformer model. The resulting effective input size to the Transformer is the same as the baseline input size. $\mathbb{T}(\cdot)$ is a Transformer model that learns from the current sentence $\mathbf{u}_i$.

Below, we will describe how to model input data embedding ϵ (Section 3.1), reservoir $\mathbb{R}$ (Section 3.2), combination operator $\uplus$ (Section 3.3), Transformer $\mathbb{T}$ (Section 3.4), and training (Section 3.5).

3.1 Embedding ϵ

First, we use the Transformer model to embed the input sentences. For a given input sentence $\mathbf{u}$, each of the J tokens w_1^J are fed to the embedding layer (Equation 3) as input and we get the sentence embedding matrix $\epsilon(\mathbf{u})$ as the output.

$$\epsilon(\mathbf{u}) = \epsilon(w_1) \oplus \cdots \oplus \epsilon(w_J) \oplus (w_{CLS}) \quad (3)$$

Here, $\epsilon(w_j)$ denotes the output of the embedding layer for the token w_j, obtained from the final hidden layer of the Transformer in the previous iteration (see STM in Section 3.4).

Since our focus is on sequence classification tasks, a special classification token "CLS" (w_{CLS}) is incorporated during the embedding process. The CLS token has been widely used in sequence classification models such as BERT (Devlin et al., 2019), ModernBERT (Warner et al., 2024), BART (Lewis et al., 2019), and DeepSeek (DeepSeek-AI et al., 2025). The CLS token is prepended to the input tokens (e.g., [CLS], sentence tokens) to capture a sentence-level representation. At each self-attention layer, it attends to all other tokens, aggregating contextual information. After the final layer, the CLS token's hidden state $\mathbf{h}_{[\text{CLS}]}^{(L)}$ summarizes the entire input and is fed into a classifier The loss function trains the CLS token to capture task-specific context such as sentiment or intent.

3.2 LTM $\mathbb{R}$

The LTM module processes all input sentences in the corpus and produces reservoir states. We use the Leaky Integrator reservoir (LI-reservoir) model (Jaeger et al., 2007), a variant of the basic reservoir, in which leaky integrator reservoir units are adopted:

$$\mathbf{x}_t = (1-\alpha)\mathbf{x}_{t-1} + \alpha \tanh(\mathbf{W}_{in}\mathbf{h}_t + \boldsymbol{\theta} + \mathbf{W}\mathbf{x}_{t-1}), \quad (4)$$

The parameters are defined as follows:

- **Reservoir State**: The reservoir state vector at time t is denoted as $\mathbf{x}_t \in \mathbb{R}^{N_r}$, capturing the network's memory of past inputs. The initial state $\mathbf{x}_0$ is either set to zero or initialized randomly. The weight matrices associated with the reservoir are initialized randomly and remain fixed throughout training. The leaky integration parameter $\alpha \in [0, 1]$ and the spectral radius ρ are optimized on the validation set using Powell's algorithm.
- **Activation Function**: The nonlinearity in the reservoir is introduced via the element-wise hyperbolic tangent function, $\tanh(\cdot)$.
- **Input and Output Dimensions**: N_u is the number of input units, and N_r is the number of reservoir (hidden) units.
- **Leaky Integration and Hyperparameters**: The leaky integration parameter α regulates the memory of the reservoir; smaller values favor longer memory. The spectral radius ρ is scaled to satisfy the Echo State Property (ESP) (Jaeger and Haas, 2004; Tiňo et al., 2007), and the sparsity level is configured accordingly. These hyperparameters are tuned on the validation set.
- **Weight Matrices**:
  - $\mathbf{W} \in \mathbb{R}^{N_r \times N_r}$ is the fixed, recurrent reservoir weight matrix, generated randomly with elements drawn independently from a Gaussian distribution.
  - $\mathbf{W}_{in} \in \mathbb{R}^{N_r \times N_u}$ is the input-to-reservoir weight matrix, where each element is sampled uniformly from $[-\sigma_{in}, \sigma_{in}]$, with σ_{in} as the input scaling factor.
  - $\boldsymbol{\theta} \in \mathbb{R}^{N_r}$ is the bias-to-reservoir vector, also initialized from the same distribution.

  These matrices are not updated during training; instead, they remain fixed at their initial random values.

Nonlinear Readout: In reservoir computing, the readout component is responsible for mapping reservoir states to the output space. Rather than using a conventional linear readout, we employ a single-layer multilayer perceptron (MLP) to compute the output at each time step t:

$$\mathbf{h}_t = \sigma\left(\mathbf{W}_{out}\mathbf{x}_t + \boldsymbol{\theta}_{out}\right), \quad (5)$$

where $\mathbf{h}_t \in \mathbb{R}^m$ is the readout with dimensionality m, $\mathbf{W}_{out} \in \mathbb{R}^{m \times N_r}$ is the reservoir-to-readout weight matrix, and $\boldsymbol{\theta}_{out} \in \mathbb{R}^m$ is the bias vector. The activation function σ is chosen to be ReLU.

Integrating a nonlinear readout into reservoir computing significantly boosts the model's expressiveness, enabling it to capture complex patterns that linear readouts cannot. This leads to improved performance on tasks with nonlinear input–output relationships, such as classification and sequence modeling. Moreover, using a nonlinear readout

improves task performance without the need to increase the size or complexity of the reservoir itself, making it a cost-effective enhancement. It also enables greater flexibility in adapting to different output structures and supports modern training techniques, such as gradient-based optimization, when appropriate. Overall, the nonlinear readout serves as a powerful and efficient extension to the reservoir architecture, yielding improved accuracy and generalization.

Group reservoirs: As ensemble reduces the variants of the prediction for a more reliable output, we integrate multiple reservoirs' nonlinear readout layers for the stability of our prediction. We consider L reservoirs, each with distinct decay rates (α and ρ), initialized randomly (Gallicchio et al., 2017) and concatenate $\oplus$ each reservoir's output to get the group reservoirs' output $\mathbf{o}_t$:

$$\mathbf{o}_t = \mathbf{h}_t^1 \oplus \mathbf{h}_t^2 \oplus \ldots \oplus \mathbf{h}_t^L \tag{6}$$

Thus, the reservoir computing function is

$$\mathbf{o}_t = \mathbb{R}(\mathbf{h}_{1:t}) \tag{7}$$

3.3 Combination $\uplus$

We use a cross-attention block $\uplus$ to combine the group reservoir output $\mathbf{o}_t$ from Equation 7 and the embeddings $\epsilon(\mathbf{u}_t)$ to feed into STM in Section 3.4:

$$\begin{aligned} \epsilon(\mathbf{u}_t) \uplus \mathbf{o}_t &= \mathcal{F}(\text{softmax}\left(\mathcal{K} \cdot (\mathbf{o}_t W^V)\right) + \mathbf{o}_t) \\ \mathcal{K} &= \frac{(\epsilon(\mathbf{u}_t)W^Q)(\mathbf{o}_t W^K)^T}{\sqrt{d_k}} \end{aligned} \tag{8}$$

We use $\mathcal{K}$ to denote the key, query, and value component. W^Q, W^K, W^V matrices are learnable, with d_k as an indication of the dimensions of the key. We initialize them randomly and train them together with the STM in Section 3.4. Besides, $\mathcal{F}$ is the layer norm of the feedforward neural networks with two layers of linear transformation with 768 neurons, and one layer of ReLU function in between, corresponding to the layer normalization operation and feed forward operation proposed in Transformer paper (Vaswani et al., 2017).

Here is an example to elaborate on how the reservoir outputs are paired with the embedded inputs. The reservoir state for time step 1 is $\mathbb{R}(\mathbf{h}_1)$, and the reservoir state for time step 2 is updated by reading the second time step values $\mathbf{h}_2$ into $\mathbb{R}(\mathbf{h}_1)$ to generate $\mathbb{R}(\mathbf{h}_1^2)$, and the reservoir state for time step 3 is updated by reading the third time step values $\mathbf{h}_3$ into $\mathbb{R}(\mathbf{h}_1^2)$ to generate $\mathbb{R}(\mathbf{h}_1^3)$. Now, say that we set $k = 2$, we can make the sequence classification y_3 based on the cross attention between $\mathbb{R}(\mathbf{h}_1^3)$ (with an output size of $k = 2$) and $\mathbf{h}_2^3$ shown in Equation 8. For the next time step, we read the h_4 value into the $\mathbb{R}(\mathbf{h}_1^3)$ and generate $\mathbb{R}(\mathbf{h}_1^4)$. By using the cross attention between $\mathbb{R}(\mathbf{h}_1^4)$ and $\mathbf{h}_3^4$, we can make the latest sequence classification y_4.

3.4 STM

We use Transformer (Vaswani et al., 2017) to implement STM. Note, that the STM's only inputs are embeddings, initialized using ModernBERT (Warner et al., 2024) with positional embeddings. After that, the LTM output states and the concatenated embeddings are fed into the STM to perform the prediction task. Equation 2 is defined as:

$$\langle y_t, \mathbf{h}_t \rangle = \mathbb{T}(\mathbf{o}_t \uplus \epsilon(\mathbf{u}_t)) \tag{9}$$

Here, $\mathbf{o}_t$ is the LTM output states from Equation 8, y_t is the predicted sequence class, $\mathbf{h}_t$ is the final layer output of the model $\mathbb{T}$, and $\mathbb{T}$ is the Transformer model. *The STM is model-agnostic and can be implemented using any neural network architecture*, such as RNNs, CNNs, LSTMs, and can be applied within encoder-only models (e.g., BERT (Devlin et al., 2018), ModernBERT (Warner et al., 2024)), decoder-only models (e.g., DeepSeek (DeepSeek-AI et al., 2025)), or encoder-decoder architectures (e.g., T5 (Raffel et al., 2023)).

3.5 Training and Parallelization

Training Loss Given $\tilde{y}_t$ as the true label and $\hat{y}_t$ as the prediction at iteration t, we use the cross-entropy loss function as our objective:

$$\mathcal{L}(\tilde{y}, \hat{y}) = -\frac{1}{T}\sum_{t=1}^{T} \tilde{y}_t \log(\hat{y}_t) \tag{10}$$

Batch Parallelization Training models with an integrated reservoir requires processing the dataset sequentially to capture the inherent memory dependencies between samples, as the computation for each input depends on the reservoir state produced by its predecessor. This sequential nature makes traditional batch training and standard Transformer parallelization impractical. To address this, we propose a novel parallelization method tailored for our

Table 1: Comparison of time and memory complexity.

Model	Time	Memory
Transformer	$O(K^2d)$	$O(Kd + K^2)$
RNN	$O(Kd^2)$	$O(Kd)$
Longformer	$O(Kd^2 + gKd)$	$O(Kqd)$
Mamba	$O(Krd)$	$O(rd^2)$
RT	$O(Kqd)$	$O((qd + q^2 + n^2)/B)$
RT(Batch)	$O(Kqd)$	$O(qd + q^2 + n^2)$

ResFormer architecture, which processes input *sentence by sentence*, while maintaining the necessary reservoir state transitions. The method proceeds in the following steps:

Batch Parallelization To support efficient training while preserving the sequential memory dynamics of the reservoir, we introduce a custom batch-parallelization strategy, as follows:

1. At the start of a new corpus, the reservoir state is randomly initialized and trained from scratch to learn the long-term dependencies inherent in the sequence.

2. Multiple sentences are grouped into a batch (e.g., batch size of 4). Each sentence is independently embedded using the ModernBERT embedding layers (Warner et al., 2024).

3. Each embedding is combined with the current reservoir state using the $\uplus$ operation, as described in Section 3.3. All sentences in the batch use the same input reservoir state, specifically, the final state from the previous batch.

4. These combined representations are processed in parallel by the STM, producing a hidden state for each sentence.

5. The resulting hidden states are then passed *sequentially*, in sentence order, through the reservoir to update its internal state. This ensures that temporal dependencies across sentences are preserved.

6. The final reservoir state, after processing the entire batch, is carried forward and used as input for the next batch.

Complexity: Table 1 shows the time and memory comparison of our method with other popular models. Here, K denotes the input sequence length, q is the sentence length (or window size in the case of LONGFORMER (BELTAGY ET AL., 2020)), g refers to the number of tokens used for global attention, d is the hidden dimension for each model, B represents the batch size, and n is the number of neurons the reservoir. Transformer has a time complexity of $O(K^2 \times d)$, which becomes impractical to compute as the input length K grows large due to the quadratic complexity.

In contrast, our ResFormer separates processing into two components: Short-Term Memory (STM) and Long-Term Memory (LTM). The LTM module operates with linear time complexity $O(K)$ relative to input length, enabling efficient handling of long contexts. While the STM retains the standard quadratic complexity of attention mechanisms, but its input length can be fixed to a constant (even a large one), allowing the overall model to scale effectively to extremely long sequences without incurring quadratic cost across the full context.

4 Experiments and Results

In this section, we present experiments demonstrating that ResFormer effectively handles contexts of arbitrary length while fully leveraging long-range dependencies.

4.1 Experimental Setup

Data and pre-processing: We present the results on four sequence classification datasets: Multimodal EmotionLines Dataset (MELD) (Poria et al., 2019), Multi-Domain Wizard-of-Oz (MultiWOZ 2.2) (Zang et al., 2020), EmoryNLP (Zahiri and Choi, 2017), and the IEMOCAP multimodal dataset (Busso et al., 2008). Each dataset is split into training, validation, and test sets following the setup in Park et al. (2022).

All datasets used are multimodal (text and visual/audio), but we focus solely on textual data (e.g., utterances) for training and evaluation, such as in MELD and IEMOCAP, to isolate the $\mathbb{RT}$ model's language understanding capabilities. While our approach is text-based, the model's architecture remains flexible and could be extended to multimodal inputs in future work.

Baselines: We compare our method with several Transformer-based baselines, including LONGFORMER (Beltagy et al., 2020), MODERNBERT (Warner et al., 2024), and DEEPSEEK-QWEN-1.5B (DeepSeek-AI et al., 2025), all fine-tuned with LORA (Hu et al., 2021). For all models, we use a weight decay of 0.01, a learning rate of

Table 2: Intent Detection on Multioz 2.2.; Emotion classification on EmoryNLP. DSLora: DeepSeek1.5B+Lora. Mem: Memory in GB. Time: Training time in hours.

Model	Accuracy	F1	Mem	Time
Intent Detection (MultiWOZ 2.2)				
DSLora	0.822	0.724	30	32
ModernBERT	0.841	0.757	15	22
RT	0.840	0.763	8	46
Emotion Classification (EmoryNLP)				
DSLora	0.310	0.256	15	3
ModernBERT	0.316	0.252	10	3
RT	0.379	0.328	3.6	6
Emotion Detection (MELD)				
DSLora	0.513	0.364	22	8.37
ModernBERT	0.516	0.377	13	7.19
Longformer	0.482	0.355	2.5	2
LSTM alone	0.326	0.287	2.1	1.33
LSTM+Reservoir	0.394	0.311	0.81	2.23
RT	0.557	0.529	4.4	10
Emotion Detection (IEMOCAP)				
DSLora	0.411	0.425	5.6	3.9
ModernBERT	0.413	0.431	5.6	4
RT	0.437	0.457	1.87	8
RT (Batch)	0.431	0.442	5.4	4

0.0002, a batch size of 16, and set the LoRA (Hu et al., 2021) α parameter to 8.

Model Training: ModernBERT (Warner et al., 2024) serves as the Transformer backbone in our model. We set the attention dropout to 0.1, weight decay to 0.01, and learning rate to 0.0002. For the LTM component, we utilize five distinct reservoirs, each initialized with different hyperparameter configurations. Specifically, the reservoir sizes range from 1500 to 1900, spectral radii from 0.7 to 0.9, leaky integration values from 0.48 to 0.52, and sparsity levels from 0.4 to 0.6. These values are optimized on the validation set using the power iteration method.

Results: Table 2 shows the accuracy (in percentage) of our ResFormer compared to several baseline models. Across all three tasks, our model consistently achieves strong performance.

- **Intent Detection:** On the MultiWOZ dataset, ResFormer performs comparably to ModernBERT (Warner et al., 2024). While it requires slightly longer training time, it is significantly more memory-efficient—consuming only about one-third of the RAM used by both ModernBERT and DeepSeek-Qwen-1.5B (DeepSeek-AI et al., 2025).
- **Emotion Classification:** On EmoryNLP (Zahiri and Choi, 2017), ResFormer outperforms all baselines by up to +6% in prediction accuracy. As with the Intent Detection task, it maintains a much lower memory footprint, requiring only one-third of the RAM used by the other models.
- **MELD and IEMOCAP:** On MELD (Poria et al., 2019), ResFormer achieves approximately a +4% improvement in accuracy over baselines. On IEMOCAP (Busso et al., 2008), it improves accuracy by +2.6% over DeepSeek and +2.4% over ModernBERT.

Despite its relatively longer training time, ResFormer delivers substantial memory savings, using only about one-third of the RAM compared to other models. Furthermore, as detailed in Section 3.5, our custom batching strategy significantly reduces training time, bringing it in line with baseline models while still achieving superior accuracy.

Compatibility with Other Backbones: The ResFormer framework is architecture-agnostic and can be integrated with non-Transformer models such as LSTMs. For example, on the MELD dataset, incorporating the reservoir into a bidirectional LSTM increases accuracy from 32.61% to 39.4% (Table 2). This highlights the reservoir's effectiveness in enhancing temporal modeling across a variety of backbone architectures.

4.2 Ablation Study

Leaky Parameter: We investigate how different leaky parameter values (α) in the reservoir affect model performance. The leaky parameter controls how much past information is retained in the reservoir state. Figure 3 shows the performance with different α values ranging from 0.3 to 0.7 on the EmoryNLP dataset. We observe that $\alpha = 0.4 \sim 0.5$ achieves the best performance, suggesting that moderate memory retention works better than either very short or long memory spans.

Nonlinear Readout: Figure 4 compares the performance of various activation functions in the reservoir readout layer on the EmoryNLP dataset. The tested activations include Linear, Tanh, ReLU, and Leaky ReLU. While ReLU consistently achieves the best results, the overall improvement from using nonlinear activations is modest.

Reservoir Size We investigate the impact of reservoir size on model performance, as it determines the dimensionality of the internal state and

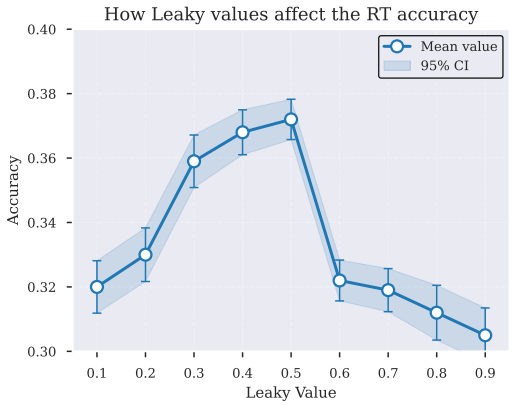

Figure 3: Leaky Values vs. model performance.

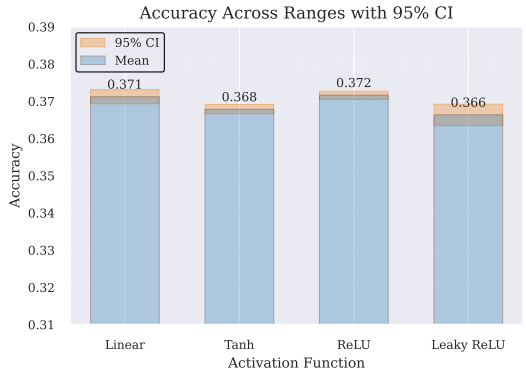

Figure 4: Activation Function vs. model performance.

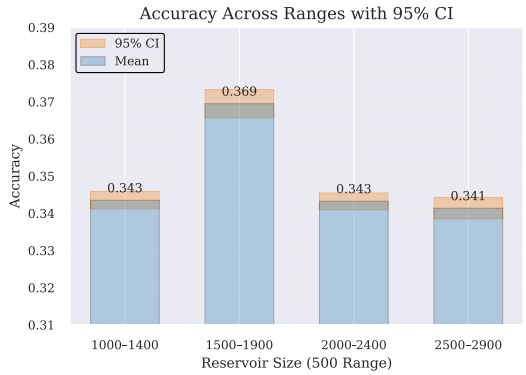

Figure 5: Reservoir size vs. model performance.

its capacity to capture temporal dependencies. Figure 5 shows results on EmoryNLP for reservoir sizes ranging from 1000 to 3000 neurons. To effectively analyze this, we use five reservoirs per model, each differing by 100 neurons (e.g., 1000, 1100, 1200, 1300, 1400), allowing us to identify which size range yields the highest accuracy.

Table 3 details the reservoir sizes within the group reservoir and their corresponding hyperparameters.

Combination Method Comparison: Ablation on EmoryNLP (Table 4) shows that replacing our cross-attention with simple concatenation or element-wise addition drops accuracy drastically (17.3% and 14.6% vs. 37.9%). This confirms cross-attention as a crucial design choice for effectively integrating reservoir states with Transformer layers, validating its role in enhancing model performance.

Table 3: Hyperparameters for each reservoir, identified by its ID.

ID ID	Reservoir Size	Spectral Radius	Leaky Values	Sparsity
#1	1500	0.9	0.48	0.6
#2	1600	0.85	0.49	0.55
#3	1700	0.8	0.50	0.5
#4	1800	0.75	0.51	0.45
#5	1900	0.7	0.52	0.4

Table 4: Combination methods between reservoir states and Transformer layers on EmoryNLP dataset.

Combination Method	**Accuracy**
Simple Concatenation	0.173
Element-wise Addition	0.146
Cross-Attention (Ours)	**0.379**

Table 5: Out-of-domain evaluation results.

Method	**Training**	**Test**	**Accuracy**
ModernBERT	EmoryNLP	MELD	0.249
ResFormer	EmoryNLP	MELD	**0.286**

Out-of-Domain Generalization: To evaluate ResFormer's robustness beyond its training domain, we conduct cross-dataset tests by training on EmoryNLP and evaluating on MELD. Due to different emotion labels, we manually aligned semantically similar categories (e.g., *Fear*/*Scared*, *Joy*/*Joyful*) to enable fair comparison. As Table 5 shows, ResFormer outperforms ModernBERT in this setting (28.6% vs. 24.9%). These results highlight ResFormer's potential for zero- or few-shot learning and suggest that the reservoir-based design supports cross-domain generalization.

ResFormer Stability: We test ResFormer with different random seeds and spectral radii across multiple datasets. As shown in Table 6, performance variance remains consistently low (e.g., 0.00025 on EmoryNLP), indicating stable training and robustness to initialization.

Training Efficiency in No-Batch Setting: We also evaluate ResFormer's efficiency on MELD by training RT and all baseline models to a setting accuracy level with a batch size of 1, demon-

Table 6: Performance variance of ResFormer across datasets with different random seeds.

Dataset	Variance
EmoryNLP	0.00025
MELD	0.00046
IEMOCAP	0.00057
Multi_OZ	0.00031

Table 7: Training efficiency and memory usage comparison on MELD (batch size = 1).

Method	ResFormer	ModernBERT	DeepSeek
Time	4 hrs	7 hrs	8.75 hrs
Memory	4.4 GB	4.4 GB	5.6 GB

strating that it outperforms both ModernBERT and DeepSeek in training time and memory usage. ResFormer reaches the setting accuracy in 4 hours, compared to 7 and 8.75 hours respectively, while using similar memory to ModernBERT (4.4GB) and less than DeepSeek (5.6GB). Although ResFormer is slightly slower per sample, it converges faster, leading to overall improved training efficiency (Table 7).

5 Related Work

Extensive research has focused on efficient long-sequence modeling, including but not limited to the works of (Kitaev et al., 2020; Kim and Cho, 2020; Beltagy et al., 2020; Choromanski et al., 2020; Katharopoulos et al., 2020; Zhou et al., 2021; Guo et al., 2021; Ma et al., 2021; Hua et al., 2022; Tay et al., 2022; Bertsch et al., 2023; Liu et al., 2023; Li et al., 2023; Mohtashami and Jaggi, 2023; Ainslie et al., 2023; Bulatov et al., 2023; Martins et al., 2021; Liu and Abbeel, 2024; Munkhdalai et al., 2024; Tworkowski et al., 2024; Bertsch et al., 2024; Han et al., 2023; Mohtashami and Jaggi, 2024).

Scaling model size, as in LLAMA 3 (Meta, 2024), is a simple fix but fails for arbitrarily long contexts (Mohtashami and Jaggi, 2024; Kryściński et al., 2021). Alternatives include fixed-size context encoding (Kanerva, 1988), modified attention to highlight salient tokens (Beltagy et al., 2020; Zaheer et al., 2020; Liu and Abbeel, 2024), heuristic attention (Liu and Abbeel, 2024; Zaheer et al., 2020), and sequence compression (Peters et al., 2018; Devlin et al., 2018), which risk information loss (Li et al., 2024). Targeted memory strategies scale beyond one million tokens (Munkhdalai et al., 2024), while others offload cross-attention to external memories like k-NN (Bertsch et al., 2024) or retrieval-augmented attention (Tworkowski et al., 2024).

Architectural innovations restructure attention via block-wise (Liu and Abbeel, 2024), ring (Liu et al., 2023), or sparse mechanisms (Zaheer et al., 2020), or embed recurrence in deep models (Munkhdalai et al., 2019; Feng et al., 2024; Bulatov et al., 2022; Wang et al., 2019; Kim et al., 2018; Bulatov et al., 2023). State-space models like Mamba (Gu and Dao, 2023) maintain compact representations but still face compression bottlenecks (Li et al., 2024).

Notably, reservoir computing (Jaeger, 2001; Maass et al., 2002; Xia et al., 2023) has been integrated with deep architectures for temporal tasks (Wang et al., 2023). While promising in speech (Nako et al., 2023; Ibrahim et al., 2021) and time series (Shahi et al., 2022; Bianchi et al., 2020; Platt et al., 2022; Shen et al., 2020), it remains underexplored for textual data.

In this work, we introduce a novel framework that captures dependencies across short, medium, and long contexts using fixed-length representations, enabling efficient long-sequence modeling while preserving rich textual information.

6 Conclusion

We propose an ResFormer model that efficiently learns long input sequences. The core novelty of our approach lies in the integration of long-term and short-term memory on the model level, which enables learn extremely long context in linear time and a constant memory. This design allows the model to capture temporal dependencies across sequences, thereby significantly improving performance on downstream tasks.

Acknowledgement

We gratefully acknowledge partial support from the eBay eRUPT Program and ACC-New Jersey (Contract No. W15QKN-18-D-0040), whose invaluable support has been essential to this work. We also thank Abdul Rafae Khan for his initial efforts in this line of research.

7 Limitations

ResFormer offers limited benefits for short sequences or tasks with minimal long-range dependencies. While full-attention Transformers may outperform it under unlimited resources, Res-

Former excels in realistic, resource-constrained settings by efficiently handling long inputs.

8 Ethical Considerations

We use AI to enhance grammar, conducting experiments exclusively on public, non-sensitive datasets. Our goal is to advance efficient and accurate NLP while promoting secure and responsible model.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Joshua Ainslie, Tao Lei, Michiel de Jong, Santiago Ontañón, Siddhartha Brahma, Yury Zemlyanskiy, David Uthus, Mandy Guo, James Lee-Thorp, Yi Tay, et al. 2023. Colt5: Faster long-range transformers with conditional computation. *arXiv preprint arXiv:2303.09752*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. 2024. Unlimiformer: Long-range transformers with unlimited length input. *Advances in Neural Information Processing Systems*, 36.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew R Gormley. 2023. Unlimiformer: Long-range transformers with unlimited length input. *arXiv preprint arXiv:2305.01625*.

Filippo Maria Bianchi, Simone Scardapane, Sigurd Løkse, and Robert Jenssen. 2020. Reservoir computing approaches for representation and classification of multivariate time series. *IEEE transactions on neural networks and learning systems*, 32(5):2169–2179.

Aydar Bulatov, Yuri Kuratov, Yermek Kapushev, and Mikhail S Burtsev. 2023. Scaling transformer to 1m tokens and beyond with rmt. *arXiv preprint arXiv:2304.11062*.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. 2022. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, et al. 2008. IEMOCAP: Interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42(4):335–359.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. 2020. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep

bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Leo Feng, Frederick Tung, Hossein Hajimirsadeghi, Mohamed Osama Ahmed, Yoshua Bengio, and Greg Mori. 2024. Attention as an rnn. *arXiv preprint arXiv:2405.13956*.

Claudio Gallicchio, Alessio Micheli, and Luca Pedrelli. 2017. Deep reservoir computing: A critical experimental analysis. *Neurocomputing*, 268:87–99.

Daniel J Gauthier, Erik Bollt, Aaron Griffith, and Wendson AS Barbosa. 2021. Next generation reservoir computing. *Nature communications*, 12(1):5564.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2021. Longt5: Efficient text-to-text transformer for long sequences. *arXiv preprint arXiv:2112.07916*.

Chi Han, Qifan Wang, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. 2023. Lm-infinite: Simple on-the-fly length generalization for large language models. *arXiv preprint arXiv:2308.16137*.

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. *Preprint*, arXiv:2106.09685.

Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. 2022. Transformer quality in linear time. In *International Conference on Machine Learning*, pages 9099–9117. PMLR.

Hemin Ibrahim, Chu Kiong Loo, and Fady Alnajjar. 2021. Speech emotion recognition by late fusion for bidirectional reservoir computing with random projection. *IEEE Access*, 9:122855–122871.

Herbert Jaeger. 2001. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13.

Herbert Jaeger and Harald Haas. 2004. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *science*, 304(5667):78–80.

Herbert Jaeger, Mantas Lukoševičius, Dan Popovici, and Udo Siewert. 2007. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural networks*, 20(3):335–352.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Pentti Kanerva. 1988. *Sparse distributed memory*. MIT press.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR.

Gyuwan Kim and Kyunghyun Cho. 2020. Length-adaptive transformer: Train once with length drop, use anytime with search. *arXiv preprint arXiv:2010.07003*.

Seungryong Kim, Stephen Lin, Sang Ryul Jeon, Dongbo Min, and Kwanghoon Sohn. 2018. Recurrent transformer networks for semantic correspondence. *Advances in neural information processing systems*, 31.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jia-Nan Li, Quan Tu, Cunli Mao, Zhengtao Yu, Ji-Rong Wen, and Rui Yan. 2024. Streamingdialogue: Prolonged dialogue learning via long context compression with minimal losses. *arXiv preprint arXiv:2403.08312*.

Shanda Li, Chong You, Guru Guruganesh, Joshua Ainslie, Santiago Ontanon, Manzil Zaheer, Sumit Sanghai, Yiming Yang, Sanjiv Kumar, and Srinadh Bhojanapalli. 2023. Functional interpolation for relative positions improves long context transformers. *arXiv preprint arXiv:2310.04418*.

Hao Liu and Pieter Abbeel. 2024. Blockwise parallel transformers for large context models. *Advances in Neural Information Processing Systems*, 36.

Hao Liu, Matei Zaharia, and Pieter Abbeel. 2023. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*.

Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. 2021. Luna: Linear unified nested attention. *Advances in Neural Information Processing Systems*, 34:2441–2453.

Wolfgang Maass, Thomas Natschläger, and Henry Markram. 2002. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560.

Pedro Henrique Martins, Zita Marinho, and André FT Martins. 2021. ∞-former: Infinite memory transformer. *arXiv preprint arXiv:2109.00301*.

Meta. 2024. Llama 3.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *arXiv preprint arXiv:2305.16300*.

Amirkeivan Mohtashami and Martin Jaggi. 2024. Random-access infinite context length for transformers. *Advances in Neural Information Processing Systems*, 36.

Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. 2024. Leave no context behind: Efficient infinite context transformers with infini-attention. *arXiv preprint arXiv:2404.07143*.

Tsendsuren Munkhdalai, Alessandro Sordoni, Tong Wang, and Adam Trischler. 2019. Metalearned neural memory. *Advances in Neural Information Processing Systems*, 32.

Eishin Nako, Kasidit Toprasertpong, Ryosho Nakane, Mitsuru Takenaka, and Shinichi Takagi. 2023. Reservoir computing system with hzo/si fefets in parallel configuration: Experimental demonstration of speech classification. *IEEE Transactions on Electron Devices*.

Hyunji Hayley Park, Yogarshi Vyas, and Kashif Shah. 2022. Efficient classification of long documents using transformers. *arXiv preprint arXiv:2203.11258*.

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

Jason A Platt, Stephen G Penny, Timothy A Smith, Tse-Chun Chen, and Henry DI Abarbanel. 2022. A systematic exploration of reservoir computing for forecasting complex spatiotemporal dynamics. *Neural Networks*, 153:530–552.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019. Meld: A multimodal multi-party dataset for emotion recognition in conversations. *Preprint*, arXiv:1810.02508.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Shahrokh Shahi, Flavio H Fenton, and Elizabeth M Cherry. 2022. Prediction of chaotic time series using recurrent neural networks and reservoir computing techniques: A comparative study. *Machine learning with applications*, 8:100300.

Sheng Shen, Alexei Baevski, Ari S Morcos, Kurt Keutzer, Michael Auli, and Douwe Kiela. 2020. Reservoir transformers. *arXiv preprint arXiv:2012.15045*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. Efficient transformers: A survey. *ACM Computing Surveys*, 55(6):1–28.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Peter Tiňo, Barbara Hammer, and Mikael Bodén. 2007. Markovian bias of neural-based architectures with feedback connections. In *Perspectives of neural-symbolic integration*, pages 95–133. Springer.

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. 2024. Focused transformer: Contrastive training for context scaling. *Advances in Neural Information Processing Systems*, 36.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tiancheng Wang, Huaping Liu, Di Guo, and Xi-Ming Sun. 2023. Continual deep residual reservoir computing for remaining useful life prediction. *IEEE Transactions on Industrial Informatics*.

Zhiwei Wang, Yao Ma, Zitao Liu, and Jiliang Tang. 2019. R-transformer: Recurrent neural network enhanced transformer. *arXiv preprint arXiv:1907.05572*.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *Preprint*, arXiv:2412.13663.

Ji Xia, Junyu Chu, Siyang Leng, and Huanfei Ma. 2023. Reservoir computing decoupling memory–nonlinearity trade-off. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(11).

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.

Sayyed M. Zahiri and Jinho D. Choi. 2017. Emotion detection on tv show transcripts with sequence-based convolutional neural networks. *Preprint*, arXiv:1708.04299.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115.