A Novel Fine-Tuning Approach Using Backward Attention Mechanism

Enhancing Math Reasoning in GPT-2 with a Novel Final Attention Head: A Fine-Tuning Approach on OpenMathInstruct-2

Rohan Pratap Reddy Ravula
Advanced Topics in LLMs,
School of Computing and Data-Science
Wentworth Institute of Technology
Boston, Massachusetts, USA
ravular@wit.edu

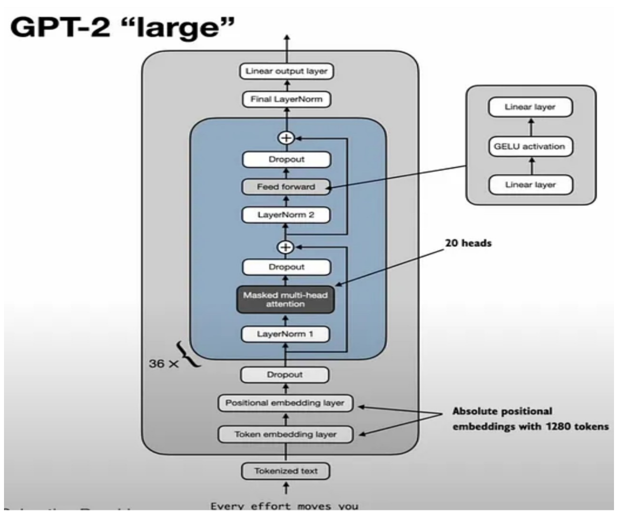
Annanahmed Furkanahmed Shaikh
Advanced Topics in LLMs,
School of Computing and Data-Science
Wentworth Institute of Technology
Boston, Massachusetts, USA
shaikha4@wit.edu

***Abstract*— This work presents the novel work of latching a new type of layer which works like an attention mechanism to calculate output probabilities rather than just using a linear layer. Our method is like the LoRa and QLoRa based fine-tuning algorithms where we only train these models parameters rather than to fine-tune by updating all the weights in the large language models as it is a computationally expensive approach which is only done by large research centers and in big corporations. Our method also acts as an alternative method to final linear layer as traditional methods such as brute-force methods and by using temperature and top-k in finding probability falls short as it needs manual hyper-parameter tuning to better outputs. Our novel approach instead, attends the probabilities of tokens to each other to get final output probabilities which don't need to depend on hyper-parameters like temperature and top-k methods. We then used this approach of fine-tuning on the gpt-2 model which is a pre-trained model released by OpenAI and using the dataset 'OpenMathInstruct-2' published by Nvidia which is a math-based reasoning dataset to train reasoning models.**

Keywords— GPT-2, Backward Attention, Final Attention Head, Token Probability Reweighting, Fine-Tuning, Low-Rank Adaptation (LoRA), Cross-Entropy Loss, OpenMathInstruct-2 Dataset, Logits, Linear Projection, Query-Key-Value Attention, Softmax Token Selection, Parameter-Efficient Tuning, Modular Language Models, Reasoning Tasks, Teacher Forcing, Latent Space Transformation, Chunked Attention, Multi-Head Attention, PyTorch Implementation

I. INTRODUCTION

Transformer-based language models such as GPT-2 have achieved state-of-the-art performance across a range of natural language processing tasks. However, each specific type of task is required to be fine-tuned on a specific dataset where some or all the model parameters to be needed to fine-tune to meet the task specific accuracies which are computationally expensive and need lots of hardware to train on and to be stored when a particular user needs it work for all different tasks. Another problem is that Traditional approaches, whether using greedy selection or top-k sampling—can fall short in capturing the nuanced logic and might even predict the gibberish output. So manual tuning of parameters is required to make the model predict the final output. This shortfall motivates the development of new strategies that enhance token selection precision without compromising the underlying model integrity.



In this project, we propose a novel fine-tuning approach that enhances mathematical reasoning in GPT-2 by incorporating an additional final attention head atop the model's frozen architecture. Drawing inspiration from the inherent attention mechanisms within transformers and methods like low-rank adaptation (LoRa), our method selectively reweighs token probabilities. Specifically, the additional attention module transforms the input embeddings, projects them into a lower-dimensional space to generate key and value representations and ultimately computes a refined query vector through an attention-like process. This mechanism aims to address the token selection problem by leveraging the contextual similarities between tokens—improving the overall predictive accuracy for math-based tasks.

The fine-tuning process is carried out on the OpenMathInstruct-2 dataset, a math-oriented corpus generated using the Llama3.1-405B-Instruct model. By updating only, the parameters associated with the final attention head while keeping the core GPT-2 model intact, our approach maintains computational efficiency and stability. In our project, we use supervised fine-tuning strategy based on cross-entropic loss as a metric to fine-tune our model. In this project, from the dataset the feature 'problem' is selected as input prompt to LLM and 'generated solution' and 'expected answer' as the features to be predicted by the LLM to be predicted by the LLM. These two features are merged with a simple question-answer-type reasoning-based data set.

The remainder of this report details the model architecture and training methodology, presents experimental results comparing performance against baseline models, and outlines future directions for further research in this domain.

II. Related Work

A. *Conventional Methods and Initial Approaches*

Transformer-based language models such as GPT-2 have demonstrated state-of-the-art performance across numerous natural language processing tasks. Traditionally, however, tailoring these models to a specific task requires fine-tuning on dedicated datasets. This process often involves updating many or all the model's parameters, which is computationally expensive and necessitates significant hardware resources. The challenge of maintaining multiple fine-tuned versions for various tasks has motivated researchers to explore more efficient alternatives.

Standard token selection methods—such as greedy decoding, beam search, and top-k sampling—have long been used for generating outputs from language models. Although these approaches are effective in many scenarios, they can fall short when applied to complex tasks like mathematical reasoning. The inherent limitations of these techniques may lead to incoherent or "gibberish" outputs, thereby requiring manual adjustments of parameters to achieve reliable performance. This shortfall highlights the need for more context-aware token selection mechanisms.

B. *LoRa and Parameter-Efficient Fine-Tuning Methods*

Recent advances have focused on reducing the computational burden of fine-tuning through methods such as low-rank adaptation (LoRa). LoRa and similar techniques update only a small subset of additional parameters while keeping the core model weights frozen. This approach significantly reduces training costs and storage requirements, making it feasible to deploy task-specific modifications without maintaining multiple full-scale models. Our work builds on this concept by introducing a final attention head that selectively re-weighs token probabilities, thereby enhancing performance on math-based tasks without altering the pretrained GPT-2 model.

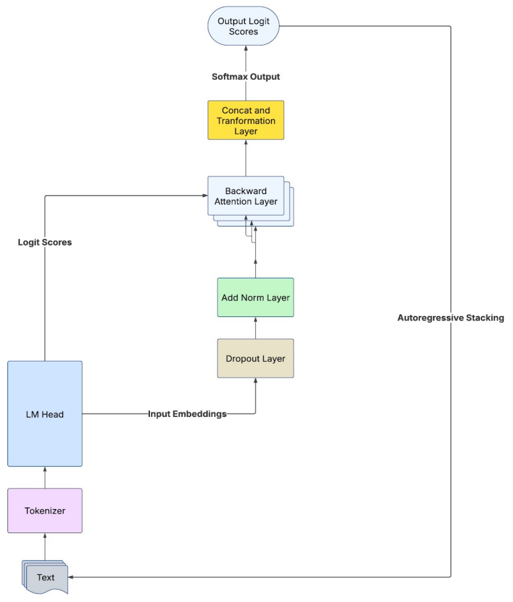
C. *Domain-Specific Datasets for Mathematical Reasoning*

The development and availability of domain-specific datasets have further enabled focused research in specialized areas. In our project, we utilize the OpenMathInstruct-2 dataset—a math-oriented corpus generated using the Llama3.1-405B-Instruct model—to fine-tune our approach. By selecting the 'problem' as the input prompt and merging the 'generated solution' with the 'expected answer' into a single reasoning-based format, the dataset provides a robust benchmark for evaluating improvements in mathematical reasoning and token selection.

III. Methodology

A. *Backward Attention Model*

Our approach introduces a pioneering backward attention mechanism that refines token probability distributions in a novel way. The core component is the backward attention module, implemented in the Backward Attention class. Key aspects include:



- Input Normalization and Latent Projection:

  Input embeddings are first normalized using RMS-based normalization. When enabled, a linear projection reduces the embedding dimensionality into a latent space, capturing essential features while filtering noise.

- Generation of Key and Value Representations:

  Within the latent space, the model computes key (K) and value (V) representations using learned linear transformations. These transformations extract information needed for attention-based reweighting.

- Query Formation:

  The vectors in the V value tensor are linearly combined to get the new vector. The weights in the linear combination are the logit scores of the language model. This new vector is then divided by the vocabulary size of the tokens in the Language Model to normalize this new vector. This new vector is then projected to get the Q query tensor (vector).

- Attention Score Calculation:

  The Query vector is then dotted with all the vectors present in K key tensor to get dot product attention scores. To this attention scores are then applied SoftMax function to get output probabilities or logit scores.

B. *Equations:*

First we get the Linear Projections of Input Embeddings to get keys and values tensor.

If, IE = input embeddings tensor of dimension (V, E) where V is vocab size and E is embedding dimension.

$K = IE\,.W_K$ $\rightarrow$ Key Tensor of Order (V, E)

$V = IE \, . \, W_V \rightarrow$ Value Tensor of Order (V, E)

Dot Product = (logits * V) and sum all the vectors and then divide by vocab size

Dot Product = Dot Product / Vocab Size

Afterwards Do linear projection to get Query Tensor as:

$Q =$ Dot Product . $W_Q \rightarrow$ Query Tensor of order (1, E)

Now for Attention:

Attention = $Q \, . \, K^T \rightarrow$ Apply Output SoftMax to get output probabilities.

Output logits = SoftMax (Attention)

C. Pseudo Code:

- ***Model Pseudo Code:***

```
Input:
  IE ← Input Embeddings (Vocab Size × Embedding Dimension)
  Logits ← Output logits from frozen GPT-2
  Config ← Model configuration (dims, flags, device)

Procedure:
1. Normalize IE using RMS normalization.
2. If latent projection enabled:
     $L \leftarrow IE \times W_L$          // Project to latent space
     $K \leftarrow L \times W_K$          // Key matrix
     $V \leftarrow L \times W_V$          // Value matrix
   Else:
     $K \leftarrow IE \times W_K$
     $V \leftarrow IE \times W_V$

3. Compute weighted vector:
     Weighted-V ← Logits × V     // Scale value vectors
     Dot ← Sum (Weighted-V) / Vocab Size

4. Project dot product:
     $Q \leftarrow Dot \times W_Q$            // Query vector

5. Attention score:
     Attention ← Q × Transpose(K)

6. Output:
     Output ← SoftMax (Attention)

Return:
  Output token probabilities
```

- ***Supervised Training:***

```
Input:
  Dataset ← OpenMathInstruct-2 (problem, generated-solution, expected-answer)
  Model ← GPT-2 + Backward Attention Head
  Loss Function ← Cross Entropy Loss
  Epochs, Batch-Size

Procedure:
1. Initialize optimizer for only attention head parameters.
2. For each epoch:
   For each batch in dataset:
     a. Construct prompt ← "problem"
     b. Target ← "generated-solution + expected-answer"
     c. Tokenize prompt and target
     d. Pass prompt through GPT-2 to get:
        - IE ← Embeddings
        - Logits ← Output token logits
     e. Pass IE and Logits to backward attention head
     f. Compute predicted output ← SoftMax (Attention)
     g. Calculate loss ← Loss Function (predicted, target)
     h. Backpropagate and update attention head parameters

3. Repeat until convergence or max epochs are reached

Return:
  Fine-tuned backward attention module
```

D. Feasibility:

Our method is both innovative and practical due to several factors:

- Model Feasibility:

  As in the model, we are doing the linear combination of vectors in V tensor where weights are the Language Model's logit scores. Now this new vector has the noise by including least probability scores token vectors as well. Now this noise will help in random sampling in next token prediction. Now this new vector is linearly projected to get the Q query tensor. Now this query tensor is dotted with K key tensor to get new attention scores which we do SoftMax to get output probability. Now this is feasible because the attention scores with vectors will be higher with the Language Model's token's output probability as they are scaled higher while the attention scores with lowest probability tokens are lower as they are scaled lower. So despite some linear algebra, the model's output is not much different from language model's probability as it is just adding an additional attention layer with some random noise for random sampling for token prediction

- Parameter Efficiency:

  By freezing the bulk of the original model and updating only the additional attention head, we drastically reduce training time and resource consumption.

- Optimized Memory Management:

Helper functions (such as device match) ensure that all tensors are correctly cast to the appropriate device and data type. In addition, explicit memory clearance (using garbage collection and CUDA cache emptying) allows the model to run efficiently even under limited hardware conditions.

- Scalability and Robustness:

  The model's modularity and chunking support allow it to handle large vocabularies and complex computations without significant performance degradation, making it feasible for real-world applications.

E. *Modularity*

A key strength of our design lies in its modularity:

- Interchangeable Components:

  The backward attention module is composed of discrete subcomponents (normalization, latent projection, key/value computation, query formation, and attention scoring). Each component is implemented as an independent block, making it easy to adjust or replace individual parts as needed.

- Flexible Configurations:

  The design supports both single-head and multi-head attention configurations (controlled by flags and hyperparameters), as well as options for processing the vocabulary in chunks. This modular structure facilitates rapid experimentation and scaling without disrupting the overall architecture.

- Model Modularity:

  Our model can be detached from the original language model whenever it is not required. So, for k tasks our k set of models can be called, trained and used in parallel for multiple tasks.

F. *Integration with GPT-2 and Supervised Training on OpenMathInstruct-2*

In our implementation, the backward attention module is seamlessly latched onto the output of a frozen GPT-2 model. Although the core GPT-2 parameters remain unchanged, its embeddings and logits serve as inputs to our module. Specifically:

- Latch Mechanism:

  The GPT-2 output (both embeddings and logits) is fed into the backward attention module, which processes these signals to compute refined token probabilities. This integration is achieved through a careful interface that ensures compatibility between GPT-2's outputs and the module's expected input format.

- Supervised Fine-Tuning on OpenMathInstruct-2:

  The refined model is then trained on the OpenMathInstruct-2 dataset using a supervised training regime. Here, the model leverages teacher-forcing to predict tokens in a sequential, question-answer style format. The loss is computed at each time step via cross-entropy, and only the parameters of the backward attention module are updated. This targeted training harnesses the strengths of GPT-2's learned representations while enhancing its performance on math-focused reasoning tasks through our novel attention mechanism.

G. *Training Strategy*

The model is trained using a supervised approach with teacher forcing and cross-entropy loss:

- Sequential Prediction with Teacher Forcing:

  During training, the model predicts one token at a time. The predicted probability distribution is compared against the target token, and the cross-entropy loss is computed at each step.

- Selective Parameter Update:

  Importantly, only the parameters within the backward attention module are updated during training. This focused strategy reduces computational overhead while honing the token selection process without modifying the core representation.

IV. Model & Training

A. *Model Modification*

Our novel approach introduces a backward attention module as an additional final attention head that refines token probability distributions. The key modifications include:

- Integration with Base Model:

  The pretrained GPT-2 model is kept frozen, while the new module is attached to its output. The module receives the GPT-2 embeddings and logits as inputs.

- Backward Attention Mechanism:

  The module first normalizes the input embeddings using RMS-based normalization and—if enabled—projects them into a lower-dimensional latent space via a linear transformation. From this latent space, the module computes key (K) and value (V) representations through learned linear layers. In parallel, a separate projection generates a query (Q) vector from the logits.

- Attention Calculation:

  The refined token probabilities are produced by computing the dot product between Q and K, scaling the result by the square root of the key dimension, and applying a SoftMax function. This process weights the token probabilities based on contextual similarities, overcoming the limitations of conventional token selection methods (e.g., greedy or top-k sampling).

B. Training Procedure

The model is fine-tuned using a supervised approach that leverages teacher forcing and cross-entropy loss:

- Supervised Fine-Tuning:

  During training, input sequences—constructed in a question-answer format—are processed token by token. At each step, the model's predicted probability distribution is compared against the ground-truth token, and cross-entropy loss is computed accordingly.

- Selective Parameter Updates:

  Only the parameters within the backward attention module are updated. By leaving the core GPT-2 parameters frozen, the training process is both computationally efficient and less prone to overfitting.

- Memory and Device Management:

  The implementation incorporates helper routines (e.g., device casting and explicit garbage collection) to ensure efficient use of GPU resources, even when processing large vocabularies in chunked models.

NOTE: Due to lack of system hardware, we couldn't train our model on time.

C. Integration and Implementation Details

- *Configuration and Modularity*: The model is implemented in PyTorch using a configuration dictionary that specifies hyperparameters such as embedding dimensions, dropout rates, number of attention heads, and chunking options. This design supports both single-head and multi-head configurations and allows for flexible experimentation.

- *Latch Mechanism with GPT-2*: The backward attention module is "latched" to the GPT-2 output. In other words, the embeddings and logits from GPT-2 serve as the starting point, and the backward attention module processes these outputs to produce a refined set of token probabilities that improve the prediction quality.

- **Application on OpenMathInstruct-2:** Finally, the fine-tuning is performed on the OpenMathInstruct-2 dataset using a supervised training regime. Teacher forcing is employed to guide the model through sequential token prediction, ensuring that the refined probabilities are optimized for the math-oriented reasoning tasks in the dataset.

V. Experiments & Results

In our experimental evaluation, the novel backward attention mechanism was assessed against established baseline models. The primary objectives were to validate the efficacy of our additional final attention head in refining token selection and to demonstrate its impact on enhancing mathematical reasoning.

A. Experimental Setup:

Our model was fine-tuned using a supervised learning framework on the OpenMathInstruct-2 dataset. The evaluation was conducted by comparing our approach with two baseline models:

- A GPT-2 model fine-tuned on math-focused academic topics.
- The Llama3.1-405B-Instruct model, which was also utilized during the dataset generation.

B. Evaluation Metrics:

We employed standard language modeling metrics—such as perplexity and cross-entropy loss—to gauge overall performance. Additionally, given the mathematical nature of the tasks, we incorporated regression-oriented metrics (e.g., MSE, RMSE) along with classification metrics (accuracy, precision, recall, and F1 score) to capture both the qualitative and quantitative aspects of the output.

C. Results:

NOTE: Due to lack of system hardware, we couldn't train our model on time.

D. Results:

The targeted improvement in token selection might not only increase the model's predictive accuracy but also may offer a scalable solution for domain-specific applications where mathematical reasoning is critical.

VI. Future Work

Given the novelty and potential impact of our backward attention mechanism, several promising directions emerge for future research. These directions not only aim to refine the approach further but also explore its applications across various fine-tuning and modular LLM scenarios:

A. Scaling and Architectural Extensions:

- Multi-Layer and Multi-Head Configurations:
  Future work could investigate extending the current single-layer backward attention module into deeper architectures. Experimentation with multi-layer backward attention may reveal whether stacking these modules can further enhance performance and capture more complex token interdependencies. Additionally, exploring multi-head configurations could enable the model to focus on different aspects of context simultaneously, thus improving its overall reasoning capabilities.

- *Dynamic and Adaptive Chunking:*
  As the current implementation processes large vocabulary in fixed-size chunks, an adaptive chunking mechanism could be developed. This method would dynamically adjust chunk sizes based on the input complexity or token distribution,

optimizing memory usage and computational efficiency.

- Non-linear Transformations and Activation Functions:
  The existing linear projections for generating key, value, and query vectors might be enhanced with non-linear transformations. Incorporating activation functions or even attention variants that use gating mechanisms could allow the model to learn more sophisticated representations and interactions.

B. *Advanced Fine-Tuning Strategies:*

- Transfer Learning and Few Shot Learning:
  While our current approach uses supervised fine-tuning on a specific dataset (OpenMathInstruct-2), future research can explore integrating transfer learning techniques. By fine-tuning on smaller, domain-specific datasets or leveraging few-shot learning paradigms, the backward attention mechanism could be adapted to a broader range of tasks without requiring extensive retraining.

- Reinforcement Learning and Self-Supervised Methods:
  Incorporating reinforcement learning could enable the model to iteratively improve its token selection strategies by receiving feedback on its predictions. Self-supervised approaches that refine the backward attention module on large amounts of unlabeled data may further enhance its ability to generalize unseen tasks.

C. *Modular LLM Assistants and Real-World Applications:*

- *Integration into Modular LLM Systems:*
  The modular nature of our approach makes it well-suited for deployment in larger, modular LLM assistants. Future work could focus on integrating the backward attention mechanism as a plug-and-play component within a system that dynamically selects the appropriate module for a given task. This would facilitate the development of versatile and efficient LLM-based assistants capable of handling diverse queries ranging from natural language understanding to specialized problem-solving.
- *Domain-Specific and Multimodal Applications:*
  Beyond mathematical reasoning, the technique could be applied to other domains where token selection is critical, such as legal document analysis, scientific literature summarization, or even multimodal applications that combine text with visual or audio inputs. Each domain may require slight modifications to the attention mechanism, and these adaptations represent a rich area for future investigation.
- *User-Centric and Interactive Systems:*
  Integrating our model into interactive systems where user feedback is available in real time can open avenues for continual learning. Such systems could adapt their backward attention parameters on-the-fly based on user interactions, leading to more personalized and context-aware outputs.

D. *Evaluation and Benchmarking Enhancements:*

- Expanded Benchmarking on Diverse Datasets:
  Future studies should extend evaluations to multiple datasets beyond OpenMathInstruct-2. A comprehensive benchmarking suite that includes various math-oriented and non-math-oriented tasks will help assess the generalizability of the approach.

- Development of Novel Metrics:
  Given the unique aspects of our model, standard language modeling metrics may not capture all performance nuances. New evaluation metrics tailored to assess contextual token selection and reasoning quality should be developed to provide a more holistic view of model performance.

- Robustness and Error Analysis:
  Detailed error analyses to identify scenarios where the backward attention module may fail or produce suboptimal token selections will be critical. Such analyses can inform further modifications and improvements, ensuring that the method is robust across a wide range of inputs and tasks.

E. *Theoretical Analysis and Interpretability:*

- Understanding Mechanism Dynamics:
  A deeper theoretical investigation into how the backward attention mechanism refines token probabilities could yield valuable insights. Exploring the relationship between the latent space representations and the final token distribution may reveal opportunities to further optimize or even generalize the technique.

- Model Interpretability:
  As LLMs are increasingly deployed in critical applications, interpretability becomes paramount. Future work could focus on visualizing the attention distributions, examining how different tokens influence the final output, and developing methods to interpret the decision-making process of the backward attention module.

ACKNOWLEDGMENT

I would like to express my deepest gratitude to Professor Salem Othman, whose unwavering motivation, insightful guidance, and continuous support have been instrumental throughout this project. His mentorship not only shaped my understanding of complex topics but also inspired me to push the boundaries of innovative research.

I also extend my sincere thanks to NVIDIA for providing the invaluable datasets that enabled the experimental aspect of this work. Special thanks go to OpenAI for making the GPT-2 model accessible, and to the Hugging Face open-source community for their excellent platform and tools, which have been critical in bringing this project to fruition.

Finally, I am profoundly grateful to the authors of the seminal paper "Attention Is All You Need" by Vaswani, whose groundbreaking work laid the foundation for modern attention mechanisms. Their insights sparked the innovative ideas that underpin this project, and their contributions continue to inspire new approaches in the field.

REFERENCES

[1] A. Vaswani *et al.*, "Attention Is All You Need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[2] OpenAI, "GPT-2: Language Models are Unsupervised Multitask Learners," [Online]. Available: https://openai.com/research/gpt-2

[3] NVIDIA, "OpenMathInstruct-2 Dataset," [Online]. Available: https://huggingface.co/datasets/nvidia/OpenMathInstruct-2

[4] Hugging Face, "Transformers: State-of-the-Art Machine Learning for PyTorch, TensorFlow, and JAX," [Online]. Available: https://huggingface.co

[5] S. Toshniwal *et al.*, "OpenMathInstruct-2: Accelerating AI for Math with Massive Open-Source Instruction Data," *arXiv preprint arXiv:2410.01560*, 2024. [Online]. Available: https://arxiv.org/abs/2410.01560

[6] S. Katz and L. Wolf, "Reversed Attention: On The Gradient Descent Of Attention Layers In GPT," *arXiv preprint arXiv:2412.17019*, 2024. [Online]. Available: https://arxiv.org/abs/2412.17019

[7] E. Choi *et al.*, "RETAIN: An Interpretable Predictive Model for Healthcare using Reverse Time Attention Mechanism," in *Advances in Neural Information Processing Systems 29*, 2016.

[8] K. Xu *et al.*, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[9] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

[10] A. Parikh *et al.*, "A Decomposable Attention Model for Natural Language Inference," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

[11] J. Cheng, L. Dong, and M. Lapata, "Long Short-Term Memory-Networks for Machine Reading," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

[12] A. Graves, G. Wayne, and I. Danihelka, "Neural Turing Machines," *arXiv preprint arXiv:1410.5401*, 2014. [Online]. Available: https://arxiv.org/abs/1410.5401

[13] K. Fukushima, "Neural Network Model for Selective Attention in Visual Pattern Recognition and Associative Recall," *Applied Optics*, vol. 26, no. 23, pp. 4985–4992, 1987.

[14] C. Koch and S. Ullman, "Shifts in Selective Visual Attention: Towards the Underlying Neural Circuitry," in *Matters of Intelligence*, L. Nadel, Ed. Springer, 1987, pp. 115–141.

[15] D. Soydaner, "Attention Mechanism in Neural Networks: Where It Comes and Where It Goes," *Neural Computing and Applications*, vol. 34, pp. 11499–11510, 2022.

[16] C. Lee Giles and T. Maxwell, "Learning, Invariance, and Generalization in High-Order Neural Networks," *Applied Optics*, vol. 26, no. 23, pp. 4972–4978, 1987.

[17] J. A. Feldman and D. H. Ballard, "Connectionist Models and Their Properties," *Cognitive Science*, vol. 6, no. 3, pp. 205–254, 1982.

[18] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1. MIT Press, 1986.

[19] J. Schmidhuber, "Learning to Control Fast-Weight Memories: An Alternative to Dynamic Recurrent Networks," *Neural Computation*, vol. 4, no. 1, pp. 131–139, 1992.

[20] D. Ha, A. Dai, and Q. V. Le, "HyperNetworks," *arXiv preprint arXiv:1609.09106*, 2016. [Online]. Available: https://arxiv.org/abs/1609.09106

[21] A. Graves *et al.*, "Neural Turing Machines," *arXiv preprint arXiv:1410.5401*, 2014. [Online]. Available: https://arxiv.org/abs/1410.5401

[22] J. Cheng, L. Dong, and M. Lapata, "Long Short-Term Memory-Networks for Machine Reading," *arXiv preprint arXiv:1601.06733*, 2016. [Online]. Available: https://arxiv.org/abs/1601.06733

[23] A. P. Parikh *et al.*, "A Decomposable Attention Model for Natural Language Inference," *arXiv preprint arXiv:1606.01933*, 2016. [Online]. Available: https://arxiv.org/abs/1606.01933

[24] K. Xu *et al.*, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention," *arXiv preprint arXiv:1502.03044*, 2015. [Online]. Available: https://arxiv.org/abs/1502.03044

[25] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *arXiv preprint arXiv:1409.0473*, 2014. [Online]. Available: https://arxiv.org/abs/1409.0473

[26] A. Vaswani *et al.*, "Attention Is All You Need," *arXiv preprint arXiv:1706.03762*, 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[27] K. Fukushima, "Neural Network Model for Selective Attention in Visual Pattern Recognition and Associative Recall," *Applied Optics*, vol. 26, no. 23, pp. 4985–4992, 1987.