Gated ResNet Architecture for Transformer Multi-head Attention Block

1st Mubtasim Fuad Mahde
Department of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
mubtasim.fuad.mahde@g.bracu.ac.bd

2nd Ahnaf Hassan
Department of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
ahnaf.hassan@g.bracu.ac.bd

***Abstract*—Transformer model architectures show excellent performance over machine learning-related tasks, primarily focusing on natural language processing by removing the need for recurrence and convolutional techniques. Residual networks (ResNet) are a core component for transformer models to retain long-term dependencies for sequence-to-sequence-related tasks. This paper introduces a new ResNet for the attention layer of a transformer model. Through a ResNet connection between layers of multi-head blocks, we flow information from one layer to the next while keeping the number of parameters the same. We also explore two-layer deep connections that retain long-term dependencies even more than one layer. Furthermore, we implement a gating mechanism on the ResNet that will selectively allow less redundant information to flow through and ensure that the gradient convergence of the model can be accelerated. In this study, we intend to prove that the performance of a model can increase even if the parameter increases due to introducing the gated unit which is insignificant in comparison to the overall number of parameters. Therefore, enhancing the model's performance on seq2seq tasks without demanding additional training time and memory by adjusting a few trainable parameters introduced for the gating mechanism is the primary goal of the paper. Our model showed promising results, especially on long-term dependent seq2seq tasks, by achieving a better performance score as well as maintaining similar efficiency.**

***Index Terms*—Transformer model, Residual networks (ResNet), Attention layer, Multi-head blocks, Long-term dependencies, Sequence-to-sequence tasks, Gating mechanism, Gradient convergence, Parameter increase, Memory reduction, Seq2seq tasks, Training stability, Sparse attention scores, Model performance, Information flow, BLEU score**

I. Introduction

The Transformers [1] model is the peak state-of-the-art (SOTA) technology when talking about language understanding and processing using neural network and machine learning. The versatility of the model in application are immense. Similarly, the model also excels in performance as well. Consequently, the model finds widespread application in various tasks such as NLP,computer vision, etc. Transformers became the building blocks for many models, including BERT and GPT. The overall architecture of transformer models is complex and difficult to interpret, so it is often termed a "black box."

The base-model transformer introduced in the study [1] contains primarily two blocks. The first block contains a stack of six identical encoders, followed by another block containing a stack of six identical decoders. For simplicity in understanding the model, let's consider input as series of word that are in a list or sequence. These sequences of words are converted to tokens and represented as continuous vectors. Position information about the sequence of words is added to the continuous vector representation and fed into the model. Inside each of the encoder, there are mainly two sub-layers. self-attention sub-layer is followed by a simple feed-forward neural network. The input sequence is replicated into multiple heads, which pass through a linear projection to create the query, key, and value matrices. Since each token of the input sequence attends to every other token in the sequence, it is termed self-attention. These matrices are computed together to get the attention score. Furthermore, each of the Q, K, and V matrices is unique to each other as the trainable parameter matrices (W_Q, W_K, and W_V) used in the linear projection are randomized. Consequently, this allows the model to capture similarities between the tokens based on different contexts by using operating a dot product multiplication between the query and key matrices. Later, the weighted sum of the scores is pushed through a simple neural network to generate an output of the attention scores. Additionally, there are ResNet connections between each sub-layer to avoid vanishing or exploding gradient problems as well as to conserve the original information of the input sequence. Unlike the encoders, the decoders contain three sub-layers. Firstly, the masked multi-head self-attention sub-layer shifts the input sequence by one position and then lets the model predict the masked words, generating masked attention scores. Followed by the first sub-layer, the second sub-layer also computes attention scores, taking the query and key input from the encoder output and the value from the masked attention scores calculated beforehand. Finally, similar to the encoder layers, the third sub-layer of the decoder is also a feed-forward neural network that converts the attention scores into an output. The generated output is passed through a SoftMax function that calculates the probability of the next sequence of tokens.

A. Research Problem

It is established that transformer models revolutionized various seq2seq tasks, yet limitations such as computation cost and training efficiency are still a major challenge to

overcome. This paper will primarily emphasize increasing the training efficiency of transformer models as well as improving performance. Moreover, the increase in learnable parameters leads to higher memory requirements. Because an increase in parameters and memory requirements will decrease the performance and efficiency of the model, On the other hand, the interpretability of the model is also important. Creating complexity in the model often leads to lack of interpretability making the models behaviour unpredictable. To address these challenges, we took inspiration from Highway Transformer [2] and RealFormer [3]. Firstly, highway transformers utilize skip connections—highway networks, to be precise. By replacing the normal ResNet connections in the base transformer model, they utilize gated skip connections. As a result, the gating mechanism can selectively leave out redundant information, resulting in faster gradient convergence. According to a paper, "subsidiary content-based SDU gates allow for the information flow of modulated latent embeddings through skipped connections, leading to a clear margin of convergence speed with gradient descent algorithms" [2]. This process will accelerate the model's push towards suboptimal points during optimization. Consequently, there are drawbacks to this architecture. The gating mechanism known as SDU increases additional parameters and subsequently increases memory requirements. Accordingly increases the computational cost and training time and decreases the overall performance of the model, especially for deeper layered transformers. Secondly, RealFormer also utilizes skip connections, i.e., ResNets, in the multi-head sub-layer. The residual connections carry information about the scaled dot product attention score ($QK^T/root(D_k)$) from the previous encoder/decoder layer to the next layer.

B. Research Objective

In simple terms, our research objective is to increase model performance as well as the training efficiency of transformer models. We aim to achieve these goals by implementing three approaches that involve highway networks and ResNet connections on the scale dot-product score of the self-attention sub-layer. This paper will empirically evaluate the performance as well as training efficiency of proposed model compared to other transformer models. We will also highlight any anomalies in our results. Furthermore, implementation of the highway network containing gating mechanisms will increase learnable parameters for the model, and so optimization for memory usage is also considered one of our priorities. We evaluate the results of three approaches by comparing them with other base-level transformer architectures. We will also try to understand which behavior or nature of the model yielded such results. We will also provide suggestions on how to improve the model.

II. Related Work

The transformer model was introduced initially in the paper "Attention is All You Need," established by Vaswani et al. [1]. The translation of languages from one to another was the key purpose behind the new model. Moreover, the model utilizes the concept of "attention mechanisms to draw global dependencies between input and output" [1]. Consequently, the model was able to show "significantly more parallelization" compared to older NLP models like RNN and LSTM [1]. Consequently, it allowed the model to be "trained significantly faster than architectures based on recurrent or convolutional layers" [1]. Each encoder layer of the model contains two sub-layers, one of which is a "multi-head self-attention mechanism" and the other a "position-wise fully connected feed-forward network" [1]. The input sequences are encoded position-wise and fed to the model, and the output originated from the final encoder layer is further passed into the decoder. In addition to this, the first sub-layer of the decoder is a masked attention layer where the position of the word sequence is shifted by one. The paper [1] evaluated the model based on the BiLingual Evaluation Understudy (BLUE) score and compared it with other models. It was evident that the transformer outperformed all other models and became the new SOTA technology not just in the field of NLP but in many other computer science fields as well. Furthermore, the base-transformer model has recently evolved into a more complex and efficient model.

The advantages that gated mechanisms bring to transformers have been explored in the paper [2]. In their work, they use residual connections along with a gated mechanism in a block of transformers. The SDU gates run parallel with the multi-headed pairwise attention block and the feedforward network. This allows bridging between long time gaps with minimal time loss and an increase in a few trainable parameters. According to study [2], it is evident that shallow layers tend to focus on local regions, and each layer attends to different semantic aspects. Furthermore, these gating mechanisms are essentially activation functions that are utilized traditionally in NN. According to a study, "divergences between biological and machine learning models concern non-linear activation functions" [4] and gave a comparative study over activation functions. Furthermore, the paper explained that "logistic sigmoid neurons are more biologically plausible than hyperbolic tangent neurons, the latter work better for training multi-layer neural networks" [4]. Moreover, the paper also somewhat explains the importance of having sparser data. Sparse data is easier to understand and evaluate while maintaining memory efficiency. "if a representation is both sparse and robust to small input changes, the set of non-zero features is almost always roughly conserved by small changes of the input" [4]. Sparser data are easier to differentiate by "non-linear machinery" [4]. Skip connections that are modified with a gating mechanism are usually referred to as highway networks [5]. The paper proposed a new gating mechanism known as "Rectified Linear Unit (RELU)" [4] which yielded better results compared to sigmoid functions or hyperbolic tangent functions. In 2013, further study in activation functions gave us the Leaky RELU [6] which is an improved version of the normal RELU.

The Residual Attention Layer Transformer, also known as the RealFormer, was initially established by Google researchers [3]. The proposed model "significantly outperforms the canonical Transformer and its variants (BERT, ETC, etc.)" [3]. In this variant of Transformer, the model utilizes ResNet connections between multi-head attention layers. To extend, the model adds $(QK^T/root(d_k))$ of the previous layer to the next multi-head sublayer of an encoder before calculating the attention scores. As a result, "it does not add expensive multiplication operations; performance is expected to be comparable" with other variants of transformer models [3]. On top of that, RealFormer has sparser attention in deeper layers and a lower discrepancy across all layers, indicating that attention density is less input-dependent. The model adds no extra parameters or hyper-parameters. The paper further reports that the proposed model excels at "masked language modeling, neural machine translation, and long document modeling" [3]. Moreover, the proposed model was experimented with over different dropout rates to understand the impact of the ResNet connection over the multi-head attention layer. The analysis showed promising results and outperformed on recent state-of-the-art transformer architectures like BERT-base, ADMIN, etc.

III. Description of the model

A. Introduction

Realformer model focuses on long term dependency performance for seq2seq models and it showed promising results compared to other transformer models. Another closely related model, Highway Transformer focuses on minimizing the cost of improving long term dependency performance by improving the models efficiency. These two architectures can be combined together to potentially create a model that excels at performing in long term dependent tasks while also efficiently utilizing the potential increase in parameters and memory requirements. Residual attention from RealFormer allows it to influence the attention scores of the current attention layer by retaining the attention score from the previous layer. In order to enable attention scores of previous layers to bypass certain layers, we incorporated Highway mechanisms that allow efficient information flow across the layers.

Our models are built on the idea that each attention layer focuses on different dependencies. The immediate previous layer may focus on short-term dependencies while the attention score from two layers back may focus on long-term dependencies. Using information from both attention scores would give a more nuanced value of attention over time. However, it is also important to consider the importance of the previous layer scores and the impact of those scores in the current layer attention scores. Thus, in order to achieve a dynamic control over this impact, a highway mechanism is incorporated. A highway mechanism introduces a gate through which we pass the attention score information from the previous layers based on their importance. It allows us to be able to regulate the information flow from the previous layers.

In regions of the input sequence where the current attention scores are more important than those of the previous layers. The current layer may not require additional refinement from earlier attention scores. In this case, the gate will **close**, allowing only the most recent attention scores to be processed.

In regions of the input sequence that are dependent on distant tokens, the gate will **open wider**, allowing attention scores from the previous layers to contribute to the refinement of the current attention scores.

B. Dimension Analysis for Base Transformer Model and RealFormer Model

In this section, the mathematical model of RealFormer is explained. Since the RealFormer model doesn't use any gated mechanisms the number of parameters remains the same. To prove it, we must first understand the dimensionality of the multihead block of a base transformer. Let X be the input after positional embedding, with dimensions $[\text{sequence\_length} \times D_{\text{model}}]$ and each weight matrix for Q, K, and V has dimensions $[D_{\text{model}} \times d_k]$, where sequence_length is the input sequence length, D_{model} is the number of hidden representation size, and d_k is the .

$$X \cdot W_i^Q = Q_i \quad \text{for one head of Query,}$$

This is similar for K and V matrices as well. So the dimensions for Q, K, V can be said: $[\text{sequence\_length} \times D_{\text{model}}] \cdot [D_{\text{model}} \times d_k] = [\text{sequence\_length} \times d_k]$. Thus, we can calculate the attention scores for the base transformer model as:

$$\text{ATT\_Score} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \cdot V \tag{1}$$

For the RealFormer model as:

$$\text{ATT\_Score} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}} + \text{prev}_{(n-1)}\right) \cdot V \tag{2}$$

where:

$$\text{prev}_{(n-1)} \text{ is the } \frac{QK^T}{\sqrt{d_k}} \text{ of the previous layer.}$$

Thus, the dimensionality of $\frac{QK^T}{\sqrt{d_k}}$ is: $[\text{sequence\_length} \times d_k] \cdot [d_k \times \text{sequence\_length}] = [\text{sequence\_length} \times \text{sequence\_length}]$.

C. Our Contribution

With the help of RealFormer model and existing gated mechanism works in Transformer models, we came up with three approaches.

D. Approach 1: 2 Layer Deep Extended RealFormer

Our first approach focuses on extending the existing RealFormer model. It relies on the concept of using previous attention layer scores to influence the long-term dependencies captured by the model. Theoretically, extending the RealFormer model would allow the model to handle complex sentence structures or tasks that require long term dependencies such as machine translation or summarization better than the existing model. Therefore, the attention score for a specific layer will be calculated as follows:

$$\text{ATT\_Score} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}} + \text{prev}_{(n-1)} + \text{prev}_{(n-2)}\right) \cdot V \tag{3}$$

Here, $n-1$ and $n-2$ refer to the previous two layers of the transformer.

E. Approach 2: 3 Layer Deep Extended RealFormer

In order to compare the feasibility and the tradeoffs that come with the extension of the model, our second approach uses the previous 3 attention layer scores to calculate current attention scores. Theoretically, approach 2 should perform better than approach 1. However, it should also be noted that any extension would result in increased operations thus leading to increased computational complexity. This added complexity may also increase the training time for the model. Thus, the primary objective of these two approaches is to understand if the model is feasible despite these drawbacks. The attention score for this approach can be calculated by the following:

$$\text{ATT\_Score} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}} + \text{prev}_{(n-1)} + \text{prev}_{(n-2)} + \text{prev}_{(} \right. \tag{4}$$

Here, $n-1$, $n-2$, and $n-3$ refer to the previous three layers of the transformer, respectively.

F. Approach 3: Extended RealFormer With Gated Mechanism Like SDU from Highway Transformer

Our third approach introduces a gating mechanism inspired by "information highways" where it allows the neural networks to have paths along which information can flow across layers [7]. We use a $\tanh$ gating function that will selectively cancel out any information that has less importance for the current input region from the previous layer and add it to the current layer. Introducing a gating mechanism also increases the complexity due to the increase in parameters. The parameter increase is due to the self-gating mechanism which uses its own weights that also requires training. Training requires operations and memory, increasing the complexity of the model, therefore, increasing training time. The gating function to be used in our model is as follows:

$$T(\text{prev}_{(n-1)}) = \tanh(\text{prev}_{(n-1)} \cdot W + B) \tag{5}$$

Thus, the output that will be passed to the next layer is,

$$\text{Gate} = T'(\text{prev}_{(n-1)}) = \text{prev}_{(n-1)} \odot T(\text{prev}_{(n-1)}) \tag{6}$$

where $\odot$ is the symbol for element-wise multiplication. Therefore, the attention score for the current layer can be calculated as follows:

$$\text{ATT\_Score} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}} + T'(\text{prev}_{(n-1)})\right) \cdot V \tag{7}$$

where T is the transformed matrix after passing through the $\tanh$ activation gate. We can further extend this idea and implement it for approach 1 and 2 as well with the following equations:

For approach 1:

$$\text{ATT\_Score} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}} + T'(\text{prev}_{(n-1)} + \text{prev}_{(n-2)})\right) \cdot V \tag{8}$$

Or, for approach 2:

$$\text{ATT\_Score} = \text{Softmax}(\frac{QK^T}{\sqrt{d_k}} + T'(\text{prev}_{(n-1)} + \text{prev}_{(n-2)} + \tag{9}$$

$$\text{prev}_{(n-3)})) \cdot V \tag{10}$$

G. Visualizing the models

We can visualize the model as follows:

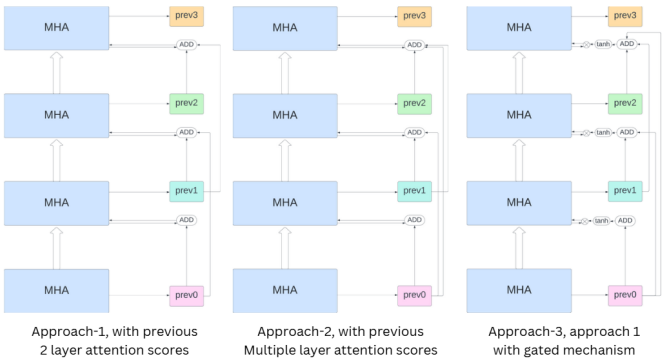

Fig. 1. Visualization of Novel Transformer Models

IV. Description of the Data

For data analysis, we are incorporating the smaller dataset, that is, the International Conference on Spoken Language Translation (IWSLT) English to Portuguese dataset [8], due to a lack of computation resources. Compared to the WMT14 English to German dataset and the WMT14 English to French dataset, the IWSLT TED-HRLr English to Portuguese (pt-en) dataset is much smaller. Which, of course, means that training our tokenizer as well as the model will take much less time. As a result, the IWSLT TED-HRLr English to Portuguese dataset is selected for preliminary analysis of the models. As for the dataset, there are a total of 52978 sentence pairs. The

source of the dataset comes from the TED Talk translation dataset, used in IWSLT evaluation campaigns, which is from the transcriptions and translations of TED Talks. These talks are public speeches on various topics, given by experts and thought leaders, which have been transcribed and translated into multiple languages by volunteers and professional translators.

A. Data analysis

The Conference on Spoken Language Translation (IWSLT) English to Portuguese dataset contains a total of 52978 sentence pairs. Out of which sentences longer than 50 words were discarded. This was done to keep the training data manageable and to avoid performance issues with very long sentences. After discarding, the new dataset contains 51965 sentence pairs. Upon further analysis, English average sentence length is 14.74, with a median sentence length of 12. Portuguese average sentence length is 13.77, with a median sentence length of 11. English language vocabulary size is equal to 27640, while Portuguese vocabulary size is equal to 39468.

Comparing the most common words in the Portuguese and

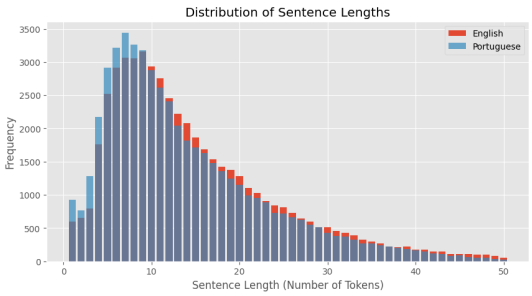

Fig. 2. Sentence Length Distribution

English word frequency distribution. We can observe that high-frequency words correspond to each other. This is in fact expected since the translation of frequent words will also be similar. For example, in Portuguese, ”a” is the feminine singular definite article, which is the most frequent word, whereas in English, ”the” is the most common. Both are articles, but in the English language there is only one definite article (”the”), whereas Portuguese has gender-specific articles, “o” for masculine and “a” for feminine, resulting in a number of high-frequency forms. When considering conjunctions, the English language “and” also corresponds to “e” in Portuguese, which means and in Portuguese. Prepositions such as ”de” (of) and ”para” (for) are used frequently in both languages. For that, they correspond to each other as expected. This demonstrates a shared linguistic pattern in the use of prepositions. The Portuguese list places less emphasis on pronouns because the language is pro-drop, with subject pronouns frequently omitted. As a result, we see the use of pronouns in English to be higher than the Portuguese translation.

V. Methodology

This section of the paper will describe the methods and setup configuration for training. The Conference on Spoken Language Translation (IWSLT) English to Portuguese datasets was used for data pre-processing, as well as model training and evaluation. And later extend this analysis over other models along with our novel architecture transformer variant that we have constructed for our machine translation task.

A. Data Prepossessing

1) Tokenization: Point to be noted: the vocabulary size of the dataset is very big and can cause performance issues when training. As a result, Byte Pair Encoding (BPE) [9] was implemented by the original transformer paper for tokenization purposes so that the vocabulary size can be maintained in a lower range, such as 50,000 or less. For our data tokenization, this paper will implement subword tokenization introduced by Google, which is also known as SentencePiece [10]. The Portuguese dataset contains a very small number of vocabulary words; hence, only 8500 top frequent words were selected as the vocabulary size of the tokenizer. This will provide us the opportunity to fully explore the dataset without having to bear high memory constaints and train the model with lower parameters.

B. Model Implementation

The model that we have implemented is the Transformer-based sequence-to-sequence translation model designed for machine translation. The dataset used for our model is the IWSLT TED-HRLr Portuguese-to-English (pt-en) dataset [8] which is widely used for machine translation. We have used a pre-tokenized dataset to speed up the training process as it avoids real-time tokenization and it also allows us to ensure consistency and efficient data pipelines during training. In order to handle the diverse vocabulary in these languages, we used a sub-word tokenizer, SentencePiece [10]. Using sub-word tokenization allows us to handle unknown words by breaking them down into smaller and more meaningful units. It also allows us to compress the vocabulary size, as uncommon words have little impact on increasing the vocabulary size.

C. Training

The model created tensors of batch size equal to 64. Set the maximum length of a sentence in terms of words to be 64 as well. The model was trained using the following hyper-parameters:

- Batch Size (`BATCH`) = 128
- Input Sequence Length (`MAX_LENGTH`) = 64
- Number of Layers (`num_layers`) = 4 — 6
- Model Dimension (`d_model`) = 512
- Feed-Forward Dimension (`dff`) = 2048
- Number of Attention Heads (`num_heads`) = 6 — 8
- Dropout Rate (`dropout_rate`) = 0.1
- Number of Epochs (`epochs`) = 16

D. Evaluation Metrics

For our experiment, this study trained a total of 15 models. The models are compared based on their validation loss and validation accuracy as well as the BLEU score.

1) Validation loss and Masked Accuracy: Two functions were implemented in our training code, one for calculating the validation loss after each epoch and another for calculating the masked accuracy after each epoch. These functions are suitable for sequential data, for instance, our machine translation task. The function ignores padded tokens and computes the loss and accuracy based on the true labels. Finally average to find the proper loss and accuracy generated. The loss is being calculated using the sparse categorical cross-entropy loss function implemented by the TensorFlow Keras library [11]. Similarly, in masked accuracy, the predicted values are compared to the true labels. Here also only the non-padded tokens are considered.

2) BLEU Score: When it comes to the sequence-to-sequence machine translation task, Bilingual Evaluation Understudy scores are often used to evaluate the translation from one language to another. A BLEU implementor's main programming task is to compare the candidate's n-grams to the reference translation and monitor the number of matches. These matches are position-independent. The more matches, the more accurate the candidate translation. To keep things simple, we'll start by computing unigram matches [12]. The original transformer paper [1] as well as the Realformer paper [3] used the BLEU score evaluation metric to compare the results of the generated output. Following their footsteps, this paper also utilized the BLEU score to compare how well the transformer model translates Portuguese text to English text.

3) Efficiency Test: Throughout the experiment, each of the models was trained on the exact hardware as well as the exact hyperparameters on the same data. The only difference would be the actual architecture of the models. This study empirically tests the training time as well as the inference time. If a model trains faster than a baseline model without losing too much performance, then it would mean that the new variant is more efficient when it comes to training. But training efficiency is not the same as the inference time. Inference time represents the time required for a model to translate a given sequence of inputs. For our experiment, we calculated the inference time on 500 sentences, each tokenized the same manner as the training dataset.

4) Sparsity Comparison: Sparsity is a key concept used when training deep networks, as it increases the efficiency of a model and utilizes less memory. It is demonstrated that the sparsity of residual blocks serves as an implicit gating mechanism for deep residual networks, avoiding the exploding or vanishing gradient problem, which is known to contribute to training difficulty [13]. This finding underscores the role of sparsity in facilitating the training of deep residual networks by mitigating issues related to gradient propagation. For our experiment we calculated how many parameters in our model are equal to or close to zero value.

VI. Experimental Analysis and Evaluation

This section of the paper highlights all the observations from the experiment. To fully understand the process and results of our observation, we need to clarify certain aspects of the training first. Below are multiple subsections on different observations filled with graphs and tables for better visualization. We primarily trained our model using the same hyper-parameters explained in our methodology section. Here we observed models' behavior across variation in number of layers as well as variation in number of heads. This provides us an insight into how the model will behave as it scales. Furthermore, the BLEU score generated by each model and the sparsity of each model are monitored. Efficiency was also calculated based on the training time, i.e., how long it took for the model to converge and saturate. Moreover, the inference time was also monitored to understand how the model would perform in real-time applications. Furthermore, to clarify the naming convention we used to report on our observation is presented here. T stands for $Transformer$ which is the vanilla baseline transformer model. R stands for the $RealFormer$ model. M stands for $Multi-Layered\ RealFormer$ where instead of working with a single previous attention score the model implements multiple previous attention scores. And so, $M2$ and $M3$ represents $Multi-Layered\ RealFormer$ with 2 previous attention scores and $Multi-Layered\ RealFormer$ with 3 previous layered attention scores respectively. Finally, we have our novel transformer variant $G1$, $G2$ and $G3$ representing the same as $Multi-Layered\ RealFormer$ with the gating mechanism. We call this model $Gated-ResNet\ Transformer$ model. For instance, $G1_E4_H8$ would mean a $Gated-ResNet$ with single ResNet connection trained with 4 layers and 8 heads.

A. Bilingual Evaluation Understudy Score

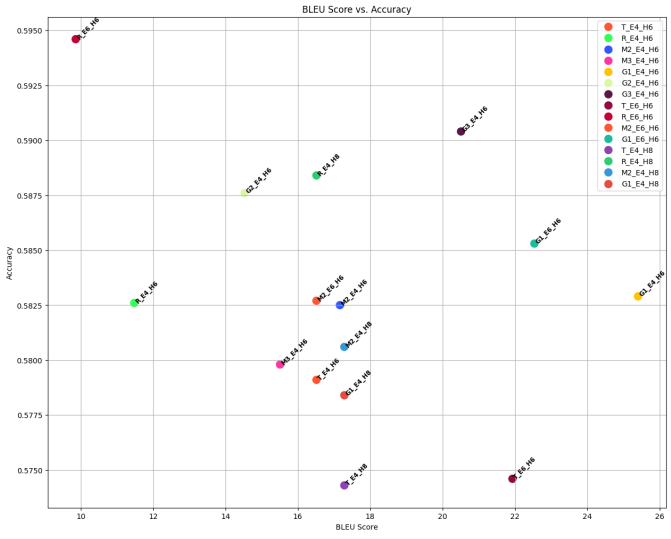

Fig. 3. BLEU Score Vs Accuracy

1) BLEU Score vs Accuracy: Here we compare the accuracy and BLEU scores of all our models. We can see that the graph is spread throughout indicating no relationship between them. A relationship between these two metrics would cause the graph to follow a linear or non-linear pattern. It should be

noted that there is one value that stands out, which is the Gated model with 1 ResNet connection, 4 layers and 6 heads. It has the highest BLEU score out of all the models even though it has an average accuracy.

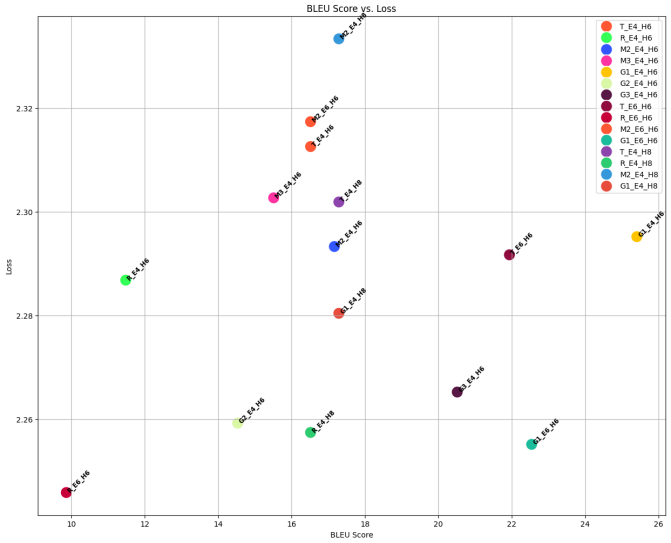

Fig. 4. BLEU Score Vs Loss

2) BLEU Score vs Loss: When we compare the loss against the BLEU score, we can see that around 7 of our models have a similar BLEU score of around 17 despite having significant differences in their loss values. While the Gated model with 1 ResNet connection, 4 layers and 6 heads has the highest BLEU score, its loss value at the end of training is average compared to the other models. The graph 4 in the page will give proper insight.

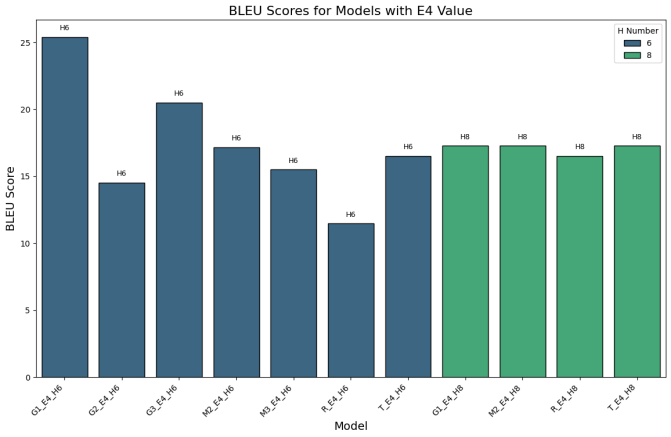

Fig. 5. BLEU Score with variable number of heads

3) BLEU Score Across Different Head parameter: For all the models with 4 encoder/decoder layers, this bar chart gives a clear view of the relationship between BLEU score and the increase in the number of attention heads. The light green indicates the models with 8 heads and blue represents the models with 6 heads. All 4 models with 8 heads resulted in a similar BLEU score indicating less impact of the architectural changes present within these models. In contrast, the models with 6 attention heads show significant differences in their BLEU score. The most significant difference can be seen in the Gated model where the model with 1 ResNet connection has almost double the BLEU score of the Gated model with 2 ResNet connections. For both Gated and Multi Layered models, it showed a decrease in the BLEU score with the increase in ResNet connections. The RealFormer model with 6 attention heads showed the lowest BLEU score out of all the models.

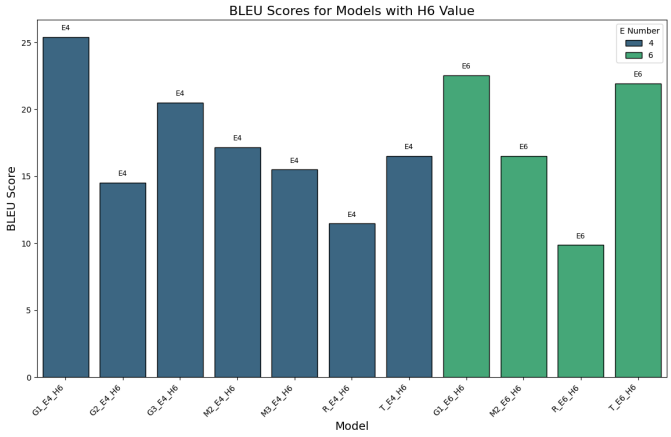

Fig. 6. BLEU score with variable number of layers

4) BLEU Score Across Different Encoder Layer: In this graph, we compare the models with 6 attention heads. The models with 6 encoder/decoder layers seemed to have a lower BLEU score compared to their respective variants with the Transformer model being an exception.

5) Overall BLEU score: The following represents a table containing the Model names with their hyper-parameter combination of number of layers and number of heads as well as the BLEU score of the corresponding models.

In this table I, we compare the BLEU scores for all our model variants. It can be seen that the RealFormer models had the minimum BLEU score despite having different configurations. The Gated models for both 4 and 6 layer configurations have the best BLEU score out of all the models. The Multi Layered models all have similar BLEU scores and do not show any significant difference from regular Transformer models.

B. Sparsity

The following table I contains the calculated sparsity of each model in percentage. The calculation was done by counting all the paramters of the model which contained values equal to zero or near zero. The counted value was divided by the total number of parameters in the models and finally represented in percentage format.

1) Sparsity Vs Training Time: As we can see in the above graph, the models show a certain trend when training time is compared to the sparsity of the model weights. The models with higher sparsity took lower time to train than the models

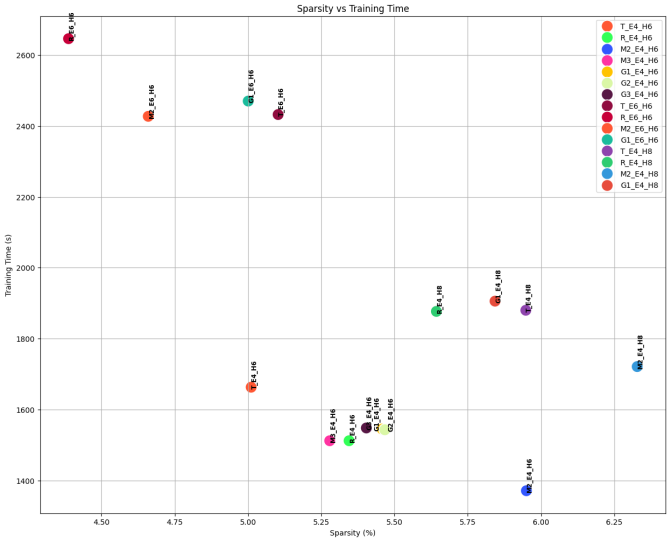

Fig. 7. Sparsity Effect on Training Time

with higher sparsity. Our Gated model that had the best BLEU score has one of the highest sparsity percentages. While the Multi Layered model with 2 ResNet connections, 4 layers and 8 heads had an above average BLEU score, its weights resulted in having the highest sparsity percentage. It can be concluded that while sparsity may have impacted training time, its relationship with BLEU score is uncertain.

C. Efficiency

Based on the above table I, we can understand the different training times taken by a model in seconds. Furthermore, we also logged the inference time. By observation of the training time, we understand that $Multi-Layered\ RealFormer$ was the fastest in training time with any combination. Unfortunately, the BLEU score is not up to the mark, and potentially the model lags significantly behind the other variants. The same goes for the $RealFormer$ model. Moreover, the inference time was calculated for translating 500 sentences from Portuguese to English. From the table we understand that our gated model with combination of 4 Encoder 6 Heads took longer time for inference compared to other varients with the same combination. We see a similar observation when it comes to the combination of 6 encoders and 6 heads.

VII. Key Findings and Discussion

A. Model Performance

Experimental observation suggests that our novel transformer variant, i.e., the $Gated\ ResNet\ Transformer$ model with a configuration of 4 encoders and 6 heads, achieved the best BLEU score, 25.406, out of all the other models. Furthermore, models with a combination of 6 Encoders and 6 Heads were also observed, and even among these combinations, the $Gated\ ResNet\ Transformer$ model achieved the peak score of 22.54. Finally, among the combinations with 4 Encoders and 8 Heads, we observe that $Transformer$, $Multi-Layered\ RealFormer$, as well as the $Gated\ ResNet$ model achieve the same BLEU score of 17.286. We did multiple tests several times to ensure that this was not a random coincidence. The score remained the same as before even after multiple tests, and we can hypothesize that this phenomenon occurred because the model is now saturated and cannot learn any more. Without increasing the encoder numbers and making the model bigger, it is difficult for it to achieve a better score. Moreover, the BLEU score that we have generated using the IWSLT English-to-Portuguese dataset is not state-of-the-art performance. A conference paper established in 2019 reported that the best score achieved in the English-to-Portuguese IWSLT dataset was approximately 26.53 [14]. Our novel variant achieved a score of 25.406, which is 1.12 less than the state-of-the-art performance score. Therefore, our novel architecture is significantly close to achieving the best score possible. By conducting future research and optimizing the model, we feel like we can achieve state-of-the-art performance. And be able to compete against state-of-the-art LLMs like the $Llama$ [15], $PaLM$ [16], $GPT-4$ [17], etc.

B. Inference Time

When it comes to real-world application, software architectures and models often depend on the inference time. The inference time refers to the total amount of time required for a model to generate an output, also known as the throughput of a model. Our $Gated\ ResNet\ Transformer$ model takes slightly higher inference time compared to the baseline vanilla $Transformer$ model. According to statistical analysis, the $Transformer$ model for combination 4 Encoders and 6 Heads takes around 1866.64 seconds to infer 500 sentences that are tokenized and sent as an input. Whereas the $Gated\ ResNet$ model takes around 2095.5 seconds. Another example, for a combination of 4 Encoders and 8 Heads, the $Transformer$ model takes 1801.8 seconds for inference, whereas the $Gated$ model takes around 1949.3 seconds. As for the combination of 6 Encoders and 6 Heads, the $Gated$ model still lags behind. Overall estimation states that the $Gated\ ResNet$ model is approximately 6.8% less efficient than the $Transformer$ model. The need to maintain high-quality outputs while adhering to tight inference time constraints, especially in safety-critical applications, is extremely important [18]. As a result, we conclude that the inference time of the model must be improved. This is due to the gating mechanism in our $Gated$ model. While inferencing, the gate weights are being sliced first, and then the attention scores are being computed. Which adds to the computation overheads. Moving forward, as for the other models, the performance of $RealFormer$ as well as the $Multi-Layered\ RealFormer$ is not up to the mark. Furthermore, the inference time is also higher than that of the $Transformer$ model. All combinations of $RealFormer$ and $Multi-Layered\ RealFormer$ performed less than the standard baseline in terms of performance, efficiency, and inference time.

TABLE I
MODEL PERFORMANCE METRICS

Model Type	Encoder	Heads	Accuracy	Loss	BLEU Score	Sparsity	Train Time (s)	Inference Time (s)
Transformer (T)	4	6	0.5791	2.3126	16.52	5.011	1663.00	1866.64
RealFormer (R)	4	6	0.5826	2.2868	11.48	5.345	1512.00	1865.60
Multi-Layered (M2)	4	6	0.5825	2.2933	17.16	5.95	1371.00	2002.60
Multi-Layered (M3)	4	6	0.5798	2.3027	15.51	5.279	1512.00	1953.50
Gated ResNet (G1)	4	6	0.5829	2.2952	25.41	5.458	1545.00	2095.20
Gated ResNet (G2)	4	6	0.5876	2.2592	14.53	5.467	1543.00	1732.10
Gated ResNet (G3)	4	6	0.5904	2.2652	20.51	5.404	1548.00	2111.40
Transformer (T)	6	6	0.5746	2.2917	21.93	5.103	2432.00	2851.20
RealFormer (R)	6	6	0.5946	2.2458	9.86	4.388	2646.00	2938.30
Multi-Layered (M2)	6	6	0.5827	2.3174	16.52	4.66	2427.00	2915.96
Gated ResNet (G1)	6	6	0.5853	2.2551	22.54	5.001	2470.00	3067.00
Transformer (T)	4	8	0.5743	2.3019	17.29	5.948	1880.00	1801.80
RealFormer (R)	4	8	0.5884	2.2574	16.52	5.643	1876.80	1970.20
Multi-Layered (M2)	4	8	0.5806	2.3334	17.29	6.328	1721.10	1834.30
Gated ResNet (G1)	4	8	0.5784	2.2804	17.29	5.843	1905.70	1949.30

C. BLEU score Behavior

For our machine translation task, we used BLEU score as the evaluation metric of our translation. This section will analyze and show if the BLEU score has any correlation with other metrics we used to evaluate the models.

1) BLEU Score Vs Accuracy: From experimental observations 3, we understand that BLEU score does not reflect any relationship with the validation masked accuracy of a model. Which implies that if the validation accuracy of a model is high, that does not necessarily translate to a better BLEU score. Furthermore, a transformer model cannot be judged and evaluated only based on the validation accuracy. Transformer models are complicated, and each model is specific to its task. As a result, each transformer model must be evaluated based on the task it has been opted for. For instance, a model's performance on a benchmark dataset is often evaluated using only a few metrics [19]. Furthermore, the single metric that is being used to evaluate our model, which is the BLEU score, may not be enough to understand the full extent and potential of our novel transformer variant. For example, when our $Gated$ mode translates the sentence "este é o primeiro livro que eu fiz" from Portuguese to English, the translation often comes as "this is the first book I did, whereas the ground truth is "this is the first book I've ever done." From this, we understand that the meaning is being portrayed to some extent, but the words and the grammar are not yet perfect. BLEU only prioritizes the words of the sentence as well as their position but lacks the ability to identify the meaning and grammar of a sentence. This shows that only a single metric is not enough to understand the behavior of the model. To conclude, BLEU scores have no significant correlation with the accuracy of the models.

2) BLEU Score Vs Loss: Similar to the relationship between BLEU score and accuracy, BLEU score also shows no correlation with the loss of a model. From the graph 4 we can easily determine that the BLEU score of the model shows no correlation with the validation masked loss of a model. For example, with the combination of 4 Encoders and 6 Heads, the least amount of validation loss was incurred by the $Gated\ ResNet\ Trasnformer$ model with 2 residual connections, which was 2.2592. Unfortunately, the BLEU score generated was only 15.51. Which actually proves that the validation loss of a model does not reflect the behavior of the BLEU score generated by the model.

D. Sparsity Vs Efficiency

Sparsity of a model contributes mostly to the training time required by the model. As the sparsity of a model increases, the number of zero values also increases in the model. This evidently helps to reduce the number of computations required by the model to converge faster. Because anything multiplied by zero is essentially set to zero automatically by the operating system. Furthermore, zero values are not stored by the hardware; as a result, less memory is required to fit the model into GPU virtual memory. The study "Low-Memory Neural Network Training: A Technical Report" looks into a variety of techniques, including sparsity, to reduce the memory footprint during training [20]. The authors show that using sparsity can result in significant memory savings with little impact on model accuracy. From the excremental graph 7 we understand that as the sparsity of the model increases, the training time required by the models also decreases. For better understanding, a best-fit line is used in the figure to explain

that the sparsity does contribute little to none to the efficiency of a model. Unfortunately, the experimental results show that this is true for all the models and not specifically the $Gated\ ResNet\ Transformer$ model. And so, the gating mechanism is not necessarily creating sparser weights or parameters in the model. For more concrete evidence, further investigation of the gating mechanism and sparsity metric is necessary. From this we can conclude that sparsity does somewhat reflect a lower training time, providing a more efficient model, but the gating mechanism is not necessarily creating the sparsity of the models.

VIII. Limitations

Throughout the entire study, oftentimes we faced multiple problems and tried to figure out a solution to those problems. In this section of the paper, we would like to address some of the limitations of our experimental study as well as a few limitations of our novel transformer variant. These limitations stem from different aspects of our study, such as methodology, resources, and generalization.

A. Computational Resources

Transformer models are termed as large language models. Typically, these models have billions of parameters, and when training the models, they utilize a huge amount of GPU virtual memory as well as computational units. Furthermore, the computational cost in terms of hardware resources scales quadratically with the sequence length/token length. And so, training transformer models became a challenging task where large input sequences cannot be used [21]. Unfortunately, due to a lack of computational resources, we were unable to conduct a more robust experiment with large transformer models. As a result, we had to shrink our models into sizable parameters just to fit the model into our GPU virtual memory. Our base model vanilla transformer had approximately 17 million parameters, whereas our own Gated ResNet model had 0.1 million more than the base transformer. Among our various models, the peak number of parameters reached was 23 million approximately. Each of the models utilized more than 20 GB of virtual memory while training. Our study had to maintain all these constraints due to a lack of computational resources.

B. Generalization Across Domains

Our novel architecture model proves to be better in terms of performance and sometimes proves to be more efficient in some cases. But the effectiveness of our gated model was only observed through a specific sequence-to-sequence task, i.e., machine translation. Ensuring that machine learning models generalize effectively across diverse domains and datasets is a critical challenge. Some of the key concerns include things like task-specific performance, limitations, and risk of overfitting specific benchmarks. Overfitting is frequently used as a generic term to describe any unintended performance drop in machine learning or transformer-based architecture models [22]. This highlights the necessity for models to be evaluated on diverse datasets as well as different tasks to ensure robust performance beyond specific benchmarks.

IX. Future Works

A. Generalization Across Domains

This study compared the novel transformer variant with other baseline transformer models, specifically in a sequence-to-sequence machine translation task. In order to understand the full potential of a novel architecture model, we need to consider experimenting with the model across multiple tasks. Deep learning applications generally assume that the training and testing distributions are identical. To accomplish this, it is critical to create models that can generalize to previously unknown distributions [23]. Not just machine translation but other tasks such as text classification, image classification, question-answer applications, chatbots, etc. Generalization across domains is crucial in machine learning and transformer models because it enables the model to be applied in different aspects of real-life problems. Furthermore, a single dataset is not optimal for comparison as the model might overfit to that certain dataset and show biased results. Therefore, machine learning models should be trained across multiple sets of datasets as well as across multiple tasks.

B. Improve Gating Mechanism

The novel transformer variant introduced in this paper utilizes a basic $tanh$ gating mechanism. Furthermore, the gating mechanism increases the complexity as well as the computation. For the gate to function, first the attention score of the previous encoder layer is multiplied with the gate weights, secondly the transformed scores are passed through a $tanh$ gate and multiplied with the previous attention score again, finally the transformed attention score is added to the current attention score. We can already observe that there are multiple arithmetic operations that are taking place. Furthermore, when considering not just adding the previous attention score from one layer but also multiple layers can also increase the computational cost. The model can be explored more by placing each individual previous score through separate gates assigned specifically for that nth previous score. This will mean that the gate weights will not be shared among all the previous layers but will increase the number of gates. Which will result in more computational cost but do have the potential to provide new observations. Last but not least, this paper used hyperbolic tangent also known as $tanh$ for the activation function used in the gating mechanism. Other types of gating functions such as the sigmoid gate, were not experimented with. For the $sigmoid$ activation function, the gradient is relatively small, particularly for inputs that are far from zero. Which can exacerbate the vanishing gradient problem in deep networks. As for $tanh$, the gradient is steeper than that of the sigmoid especially around zero. This may result in stronger gradients and eventually lead to more significant weight updates during training, which may lead to improved performance [24].

X. Conclusion

In this paper, we establish three simple and generic novel approaches for the attention layer of the transformer model aimed at improving the performance of the model as well as increasing the training efficiency. By incorporating various techniques of skip connections like ResNets and Highway Network, we facilitate the information flow of attention scores of heads from one layer to adjacent layers. Furthermore, we attain an overall performance gain with an insignificant increase in parameters and memory requirements. This paper empirically proves that the model achieves better results compared to other generic transformer models. Specifically, in sequence-to-sequence tasks, by adjusting a few trainable parameters, long-term dependencies can be maintained without requiring additional training time or memory. The novel approaches established in the paper can be incorporated with other SOTA transformer models and researched further to study the impacts of skip connections over attention heads. The model can be further tuned for specific tasks in future applications.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, vol. 30, pp. 5998–6008, 2017. [Online]. Available: https://arxiv.org/pdf/1706.03762v5

[2] Y. Chai, S. Jin, and X. Hou, "Highway transformer: Self-gating enhanced self-attentive networks," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.616

[3] R. He, A. Ravula, B. Kanagal, and J. Ainslie, "Realformer: Transformer likes residual attention," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021. [Online]. Available: https://doi.org/10.18653/v1/2021.findings-acl.81

[4] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudík, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 315–323. [Online]. Available: https://proceedings.mlr.press/v15/glorot11a.html

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. [Online]. Available: https://doi.org/10.1109/cvpr.2016.90

[6] A. L. Maas, A. Y. Hannun, A. Y. Ng *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1. Atlanta, GA, 2013, p. 3.

[7] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *arXiv preprint arXiv:1505.00387*, 2015. [Online]. Available: https://arxiv.org/abs/1505.00387

[8] M. Cettolo, C. Girardi, and M. Federico, "The IWSLT 2014 evaluation campaign: Evaluation of the spoken language translation track," in *Proceedings of the 11th International Workshop on Spoken Language Translation*, Lake Tahoe, California, USA, 2014.

[9] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *Association for Computational Linguistics*, Jan. 2016. [Online]. Available: https://doi.org/10.18653/v1/p16-1162

[10] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," *Association for Computational Linguistics*, Jan. 2018. [Online]. Available: https://doi.org/10.18653/v1/p18-1007

[11] TensorFlow, "Sparsecategoricalcrossentropy — tensorflow core v2.9.1," https://www.tensorflow.org/api_docs/python/tf/keras/losses, 2025.

[12] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu," *Association for Computational Linguistics*, Jul. 2001. [Online]. Available: https://doi.org/10.3115/1073083.1073135

[13] S. Salman and X. Liu, "Sparsity as the implicit gating mechanism for residual blocks," *2022 International Joint Conference on Neural Networks (IJCNN)*, p. 1–6, Jul. 2019. [Online]. Available: https://doi.org/10.1109/ijcnn.2019.8851903

[14] J. Niehues, R. Cattoni, S. Stüker, M. Negri, M. Turchi, E. Salesky, R. Sanabria, L. Barrault, L. Specia, and M. Federico, "The iwslt 2019 evaluation campaign," *International Conference on Spoken Language Translation*, Nov. 2019. [Online]. Available: https://cris.fbk.eu/handle/11582/321912

[15] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[16] A. Chowdhery, S. Narang *et al.*, "Palm: Scaling language modeling with pathways," 2022. [Online]. Available: https://arxiv.org/abs/2204.02311

[17] OpenAI, J. Achiam, S. Adler *et al.*, "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[18] Z. Yang, W. Gao, C. Luo, L. Wang, F. Tang, X. Wen, and J. Zhan, "Quality at the tail of machine learning inference," 2024. [Online]. Available: https://arxiv.org/abs/2212.13925

[19] K. Blagec, G. Dorffner, M. Moradi, and M. Samwald, "A critical analysis of metrics used for measuring progress in artificial intelligence," *arXiv preprint arXiv:2008.02577*, 2020.

[20] N. S. Sohoni, C. R. Aberger, M. Leszczynski, J. Zhang, and C. Ré, "Low-memory neural network training: A technical report," 2022. [Online]. Available: https://arxiv.org/abs/1904.10631

[21] B. Zhuang, J. Liu, Z. Pan, H. He, Y. Weng, and C. Shen, "A survey on efficient training of transformers," *International Joint Conference on Artificial Intelligence*, p. 6823–6831, Aug. 2023. [Online]. Available: https://doi.org/10.24963/ijcai.2023/764

[22] R. Roelofs, V. Shankar, B. Recht, S. Fridovich-Keil, M. Hardt, J. Miller, and L. Schmidt, "A meta-analysis of overfitting in machine learning," *Neural Information Processing Systems*, vol. 32, p. 9175–9185, Jan. 2019. [Online]. Available: https://papers.nips.cc/paper/9117-a-meta-analysis-of-overfitting-in-machine-learning.pdf

[23] J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, and P. Yu, "Generalizing to unseen domains: A survey on domain generalization," *IEEE Transactions on Knowledge and Data Engineering*, p. 1, Jan. 2022. [Online]. Available: https://doi.org/10.1109/tkde.2022.3178128

[24] H. Nguyen, N. Ho, and A. Rinaldo, "Sigmoid gating is more sample efficient than softmax gating in mixture of experts," *arXiv preprint arXiv:2405.13997*, 2024.