Reducing the Energy Dissipation of Large Language Models (LLMs) with Approximate Memories

Zhen Gao*, Jie Deng†, Pedro Reviriego‡, Shanshan Liu§ and Fabrizio Lombardi¶

*School of Electrical and Information Engineering, Tianjin University, Tianjin, China
†School of Future Technology, Tianjin University, Tianjin, China
‡Departamento de Ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid, Madrid
§School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China
¶Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, USA.

Email: zgao@tju.edu.cn dengjie2023@tju.edu.cn pedro.reviriego@upm.es ssliu@uestc.edu.cn lombardi@coe.neu.edu

2024 IEEE International Symposium on Circuits and Systems (ISCAS) | 979-8-3503-3099-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISCAS58744.2024.10558275

Abstract—**Large language models (LLMs) have shown impressive performance in a wide range of tasks such as answering questions or summarizing text. However, running LLMs on edge devices is challenging as they require large amounts of energy due to their memory and computation needs. In LLMs most of the memory is needed to store the model parameters which number keeps increasing from one LLM generation to the next. In the last several years, significant efforts have been made to compress and prune parameters, but this is not enough to reduce their memory needs as the number of parameters grows exponentially. In this work, to reduce energy dissipation, rather than trying to reduce the amount of memory used by LLMs, we study the use of approximate memories to store the LLM parameters. Approximate memories can significantly reduce the energy dissipation at the cost of introducing errors in some of the memory bits. Therefore, the impact of errors on LLMs must be understood. To that end, we have performed error injection on different compressed versions of a classic LLM: Bidirectional Encoder Representations from Transformers (BERT). The results show that in some cases compressed BERTs operate reliably at high bit error rates. This makes possible the use of approximate memories with a negligible impact on the LLM performance and a significant reduction in energy dissipation.**

Keywords—LLMs, approximate computing, memories

I. Introduction

Large language models (LLMs) have gained much attention for natural language processing (NLP) applications after the release of ChatGPT [1]. Since a cloud-based service usually results in large latency, many researchers and companies are starting to think about the deployment of LLMs on edge servers [2], [3]. However, the complexity and huge requirements of storage and power pose a big challenge. In the last two years, several compression/distillation and quantization techniques have been proposed for LLMs [4], [5], but this is not enough to reduce their memory needs and the corresponding energy consumption as the number of LLM parameters grows exponentially. For example, the number of weights is about 100 million for GPT-1 [6], but increases quickly to 175 billion for GPT-3 [7] and possibly trillions for GPT-4 [8] all within 3 years.

Approximate computing is a promising approach to reduce the energy dissipation of complex models [9]. In particular, approximate memories can be implemented for both SRAMs and DRAMs by reducing the supply voltage or refresh frequency [10], [11]. These techniques can significantly reduce the energy dissipation at the cost of introducing errors on the memory bits thus making the memory approximate [9]. The impact of these errors depends on the application and the importance of the bits that suffer the errors [12]. Approximate memories have been proposed for video applications [13], data sketches [14] or deep learning implementations [15] showing the potential for large power reduction for data storage and thus the overall energy dissipation. However, to the best of our knowledge there is no study of the use of approximate memories for LLMs.

In this paper, we take the Bidirectional Encoder Representations from Transformers (BERT) as an example to explore the feasibility of applying voltage scaling techniques for power saving. In particular, three compressed BERT models with two weight formats are evaluated for different memory error rates. The rest of the paper is organized as follows. Section II provides preliminaries for this study. The impact of errors in an approximate memory on the performance of three compressed BERT models with two types of weight formats are analyzed and evaluated in Sections III and IV, respectively. The potential power savings of using approximate memories for implementing LLMs are then discussed in Section V. Finally, the paper concludes in Section VI.

II. Preliminaries

A. Structure of the Standard BERT

BERT is a classic type of LLMs and the basic structure of many LLMs is similar to it. As shown in Fig. 1(a), BERT is fundamentally a stack of *N* transformer encoders [16], in which the input *X* is the addition of the Input embeddings (*I*) and the Position embeddings (*P*) for *M* words in a sentence. Each encoder consists of two sub-layers, a Multi-Head Self-Attention (MH-SA) and a Feed-Forward Network (FFN). The self-attention (SA) operates on a query matrix *Q*, a key matrix *K* and a value matrix *V* as:

$$SA(Q,K,V) = V \times softmax\left(\frac{K^T Q}{\sqrt{d_k}}\right) \quad (1)$$

where d_k is the key dimensionality, and *Q*, *K* and *V* are generated as $Q=W_QX$, $K=W_KX$ and $V=W_VX$, respectively, where W_Q, W_K, and W_V are the corresponding weight matrices. As shown in Fig. 1(b), the MH-SA mechanism obtains *H* heads, and the output of the *i*-th head is given by

$$head_i(X) = SA\left(W_Q^iX, W_K^iX, W_V^iX\right). \quad (2)$$

The outputs of all heads are merged by matrix W_O as

$$MH\text{-}SA(X) = W_O \times \text{Concat}\left(head_i(X)\right). \quad (3)$$

Then the input of the FFN is generated by residual addition with the input *X* and normalization as

$$Y = norm\left(X + MH\text{-}SA(X)\right). \quad (4)$$

FFN is composed of two fully-connected layers and a non-linear activation operation between them, whose output is

This work is supported by National Natural Science Funds of China (62171313) and the Spanish Agencia Estatal de Investigacion (PID2022-136684OB-C22).

$$FFN(Y) = W_{F2} activate(W_{F1}Y), \quad (5)$$

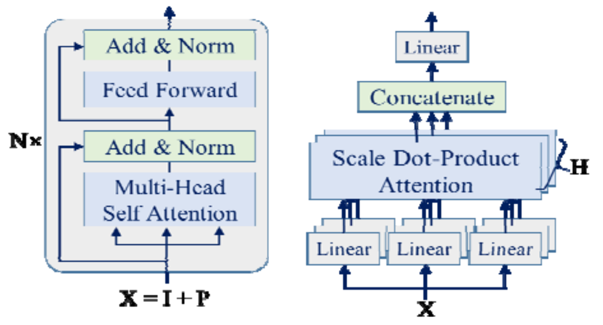
where W_{F1} and W_{F2} are the weight matrices for the two fully-connected layers, respectively. After the residual addition and normalization, the output of the encoder becomes

$$Z = norm(Y + FFN(Y)). \quad (6)$$

The standard BERT includes 12 encoders (N = 12), and each MH-SA includes 12 heads (H = 12) [16]. The dimension for each word embedding is 768, so the dimension of the input X is 768×M, and the size of the different weight matrices are listed in Table I.

TABLE I. SIZE OF DIFFERENT WEIGHTS IN THE STANDARD BERT

Type	W_Q	W_K	W_V	W_O	W_{F1}	W_{F2}
Size	768×768	768×768	768×768	768×768	3072×768	768×3072



(a) General structure of BERT (b) Structure of MH-SA

Fig. 1. Structure of BERT.

B. Compressed Versions of BERT

Many compressed models have been proposed to reduce the memory needs of BERT, among which ALBERT [17], DistilBERT [18] and Tiny BERT [19] are the most popular ones. As shown in Table II, ALBERT has the same structure as the standard BERT, but all layers share the same set of weights so the memory size is reduced by about 90%. DistilBERT reduces the number of layers by half, so that the memory size is reduced by about 40%. Tiny BERT only involves two layers with two heads in each layer; its memory size is only about 4% of the standard BERT. In this paper, these models are considered in the evaluations to investigate the use of approximate memories to further reduce the power dissipation for parameter storage.

TABLE II. STRUCTURE OF THREE COMPRESSED BERT MODELS (FLOAT16)

Model	**# of Layers**	**# of Heads**	**Hidden Dim.**	**FFN Size**	**Memory Size (MB)**	**Inference Time (ms)**
Base BERT	12	12	768	3072*768	220.0	≈ 4
ALBERT	12	12	768	3072*768	23.4	≈ 4
Distil-BERT	6	12	768	3072*768	134.0	≈ 2
Tiny BERT	2	2	128	512*128	8.8	≈ 1

C. Quantization of Model Parameters

The computation of BERTs often employs floating-point numbers such as the half-precision format (Float16, which consists of a sign bit, 5 exponent bits and 10 fraction bits) as given by

$$float16\_value = (-1)^{sign} \times 2^{exponent-15} \times (1 + fraction/2^{10}). \quad (7)$$

For efficient use of storage, quantization of model parameters can also be performed. In this case, each row of the weight matrix in Table I is quantified using Eqn. (8), where the scaler S and the zero point (Z) are defined by Eqns. (9) and (10), respecively, r is a orignal weight, r_{max} and r_{min} are the maximum and minimum of the row of float weights, q_{max} and q_{min} are the upper and lower limit of the integer value, which are (-128, 127) for INT8. During the inference, the INT8 values are recovered to the Float16 format based on Eqn. (11). Since the quantized values are potentially resilient to some errors, both Float16 and INT8 formats are considered in the evaluations of this paper.

$$q = round\left(\frac{r}{S} + Z\right) \quad (8)$$

$$S = \frac{r_{\max} - r_{\min}}{q_{\max} - q_{\min}} \quad (9)$$

$$Z = round(q_{\max} - \frac{r_{\max}}{S}) \quad (10)$$

$$r' = S(q - Z) \quad (11)$$

D. Sentence Emotion classification

In this paper, sentence emotion classification [20] is used as a case study to evaluate the impact of memory errors. In this application, a sentence can be classified into one of six emotions, including 'joy', 'sadness', 'anger', 'fear' 'love' and 'surprise'. The model is constructed by stacking a classification network (CN) on the BERT. The CN is also a FFN with two fully-connection layers; the first row of the output matrix of the last encoder of BERT is taken as the input of the CN, and the output of CN includes six scores for the six emotions. The classification result is the emotion corresponding to the highest score. In this paper, three pre-trained models from Huggingface ([21]-[23]) are used for evaluation of ALBERT, DistilBERT and Tiny BERT, respectively. The accuracy of the sentence emotion classification for the three compressed BERT models with Float16 and INT8 weights are listed in Table III. Since the accuracy of BERT base with Float16 weights is about 93%, the compressed and quantized models do not introduce obvious accuracy loss except for Tiny BERT. An NVIDIA® GeForce RTX™ 4080 GPU is used for all the experiments, and the average processing time for a sentence for the three models are listed in the last column of Table II.

TABLE III. ACCURACY OF COMPRESSED BERTS

	ALBERT	**DistilBERT**	**Tiny BERT**
Float16	93.15%	93%	90.2%
INT8	93.35%	93%	90.2%

E. Approximate Memories

To evaluate the potential power savings when using approximate memories for the compressed models, an SRAM with supply voltage scaling is considered, specifically the 28 nm SRAM with a nominal Vdd of 1.0 V presented in [24]. Such approximate memories introduce read errors (that can randomly be “1” or “0”) due to the scaled supply voltage. From the simulation results presented in [24], the power savings versus Bit Error Rate (BER) of an approximate SRAM have been computed considering the general scaling factor between power and supply voltage in CMOS technology (i.e., the dynamic power scales quadratically with Vdd, while the static power scales linearly with Vdd [24]). As shown in Fig. 2, 71.6% (46.9%) dynamic (static) power can be saved in the 28 nm SRAM by introducing a BER of 10^{-2}. In this paper, four memory BERs are considered in the evaluations presented in the following sections, including 10^{-6}, 10^{-4}, 10^{-3}, and 10^{-2}.

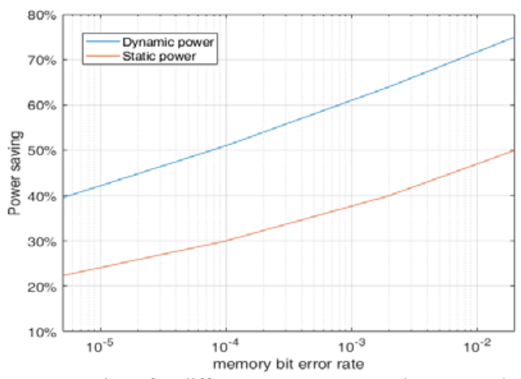


Fig. 2. Power savings for different memory BERs due to supply voltage scaling in a 28 nm approximate SRAM [14].

III. Effect of Errors on Compressed BERT with Half-Precision Parameters

A. Analysis of the Effect of Errors on Different Bits

Based on the model analysis, we found that the weights in the three compressed BERT models are normal distributed among [-0.5, 0.5], and the average value of each bit of the weight is also the same. As shown in Fig. 3, the 1st exponent bit is always 0; the 2nd exponent bit is 1 with a probability of 85%; the 3rd exponent bit is 0 with a probability over 85%; all other bits can be 0 or 1 at a probability of 50%. This means that errors on the 1st exponent bit will increase the value by 2^{16} times; errors on the 2nd exponent bit will decrease or increase the value by 2^8 times with the probability of 85% and 15%, respectively; errors on the 3rd exponent bit will increase or decrease the value by 2^4 times with the probability of 85% and 15%, respectively.

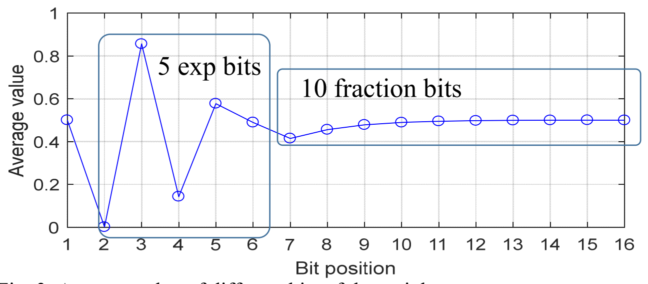


Fig. 3. Average value of different bits of the weights.

B. Impact of Errors on BERT Performance

To evaluate the impact of memory errors on BERT performance, the sensitivity of different weight bits to errors is analyzed first. This is performed by selecting several weights randomly in a matrix and flipping a specific bit (e.g., the sign bit) of each weight. The effect of errors on different bits of the Float16 weights for the three models are shown in Figs. 4, 5, and 6, respectively. Based on the results for ALBERT in Fig. 4, we can get three main conclusions: 1) for BER lower than10^{-4}, errors only introduce obvious accuracy loss when occur on the 1st exponent bit; 2) when BER increases to 10^{-3} or 10^{-2}, errors on the 2nd and 3rd exponent bits will introduce obvious accuracy loss; 3) errors on other bits have negligible impact on the model performance even for BER of 10^{-2}. The reason for 1) is that errors on the 1st exponent bit increases the weight by 2^{16}, so that the classification is fixed to a specific category due to overflow. The reason for 2) is different. Errors on the 2nd and 3rd exponent bit do not cause overflow, but many wrong weights make the results almost random. Fig. 5 presents the similar results for Distil-BERT, but the accuracy loss is obviously smaller than that for ALBERT. This is because ALBERT applies weight sharing, so the errors on a weight affect the processing of all layers. Finally, results in Fig. 6 show that errors on Tiny BERT have lower impact on the accuracy, which means the smaller model is more resilient to errors.

Next, the above experiment is repeated by flipping a random bit of each selected weight to evaluate the overall impact of random memory errors on the BERT performance. The experimental results are shown in Fig. 7. As we can see, for ALBERT and DistilBERT, BER of 10^{-4} is high enough to decrease the accuracy down to 30%. Tiny BERT provides higher error resilience, but the accuracy is also about 30% for BER higher than 10^{-3}.

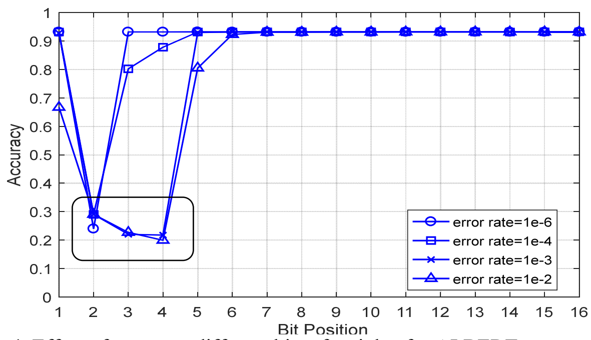


Fig. 4. Effect of errors on different bits of weights for ALBERT.

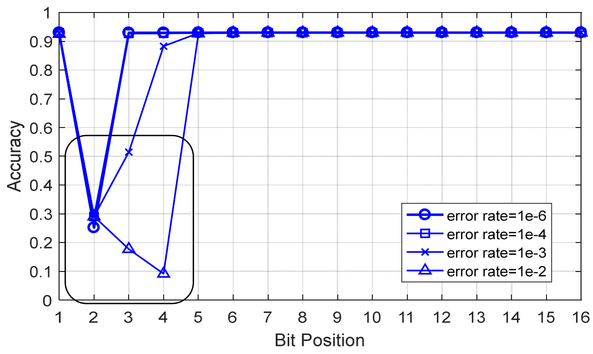


Fig. 5. Effect of errors on different bits of weights for DistilBERT.

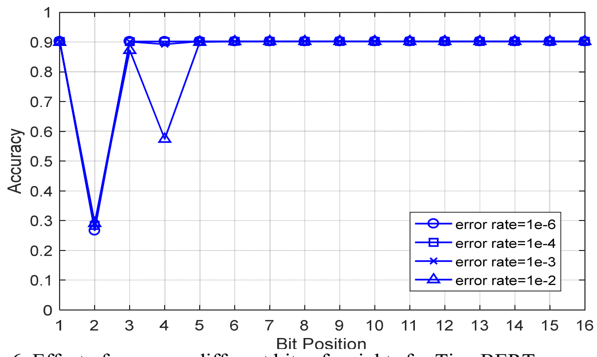


Fig. 6. Effect of errors on different bits of weights for Tiny BERT.

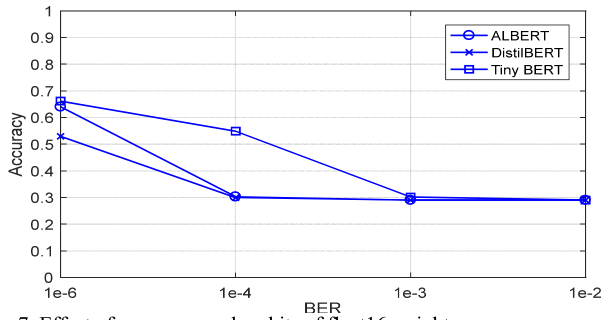


Fig. 7. Effect of errors on random bits of float16 weights.

IV. Effect of Errors on Compressed BERT with Quantized Parameters

A. Analysis of the Effect of Errors on Different Bits

In the quantized models, the INT8 weights are normal distributed among [-128, 127], and all the bits can be 0 or 1 with 50% probability. Since the flip of the n-th bit will change the INT8 weight by $\Delta = \pm 2^{8-n}$ (n = 1 for sign bit), the recovered Float16 value of the weight can be deduced as

$$r' \approx r \pm \delta / 2^n \tag{12}$$

where $\delta = r_{max} - r_{min}$, and is smaller than 1 for most cases. In this case, there is no critical bit that causes overflow as for Float16 weights, but higher bits (smaller n) will have larger impact to the model performance.

B. Impact of Errors on BERT Performance

Again, the impact of errors on different bits of the INT8 weights for the three models is initially evaluated using the same error injection approach described in Section III-B. As per the results shown in Figs. 8 to 10, in general, errors on higher bits have larger impact on the accuracy. Among the three models, errors have more obvious impact on ALBERT due to weight sharing, and the performance of Tiny BERT is almost unaffected by the errors even for BER of 10^{-2}. These results are consistent with the analysis in Section IV-A.

Fig. 11 shows the impact of errors on random bits of INT8 weights. Only errors with BER of 10^{-2} are shown to introduce obvious accuracy loss due to weight sharing, and there is negligible accuracy loss for other cases.

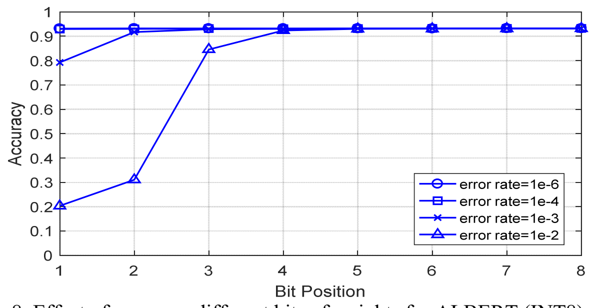


Fig. 8. Effect of errors on different bits of weights for ALBERT (INT8).

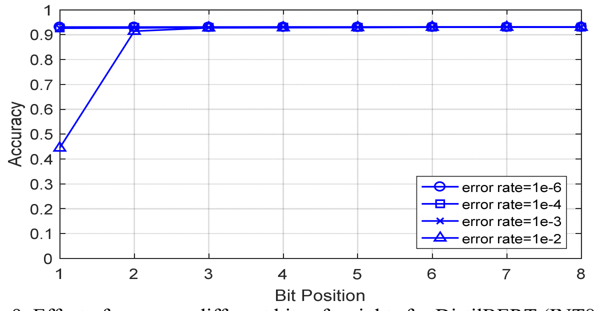


Fig. 9. Effect of errors on different bits of weights for DistilBERT (INT8).

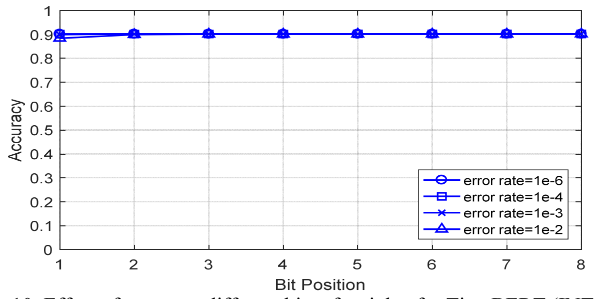


Fig. 10. Effect of errors on different bits of weights for Tiny BERT (INT8).

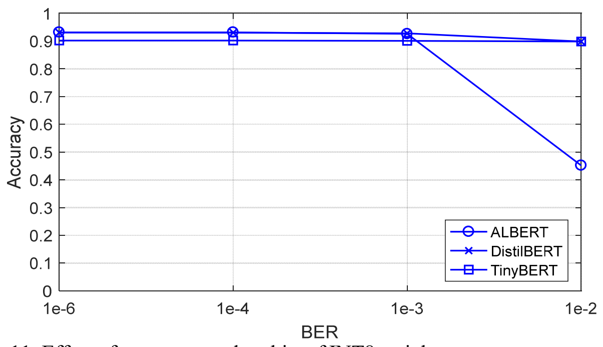


Fig. 11. Effect of errors on random bits of INT8 weights.

V. Power Saving of LLMs with Approximate Memories

From the results in the previous sections, the following observations can be obtained. First, for compressed BERTs with half-precision parameters, it is not feasible to employ an approximate memory by voltage scaling, because the memory errors significantly degrade the model accuracy (even if with a low BER) as shown in Fig. 7.

However, for compressed BERTs with quantized parameters, there is an opportunity to reduce power dissipation by storing the parameters in an approximate memory. Specifically, from Fig. 11, DistilBERT and Tiny BERT can almost maintain the error-free accuracy at a BER up to 10^{-2} and ALBERT up to 10^{-3}; as per Fig. 2, the power savings in these cases are given in Table IV. It can be seen that the use of approximate memories in the case of parameters represented with the INT8 format are significant and will contribute to make the implementation of LLMs on edge devices possible in the future.

TABLE IV. Potential Power Savings for INT8-based BERTs with Approximate Memories

	ALBERT	DistilBERT	Tiny BERT
Static	Up to 37.6%	Up to 46.9%	Up to 46.9%
Dynamic	Up to 60.9%	Up to 71.6%	Up to 71.6%

VI. Conclusion

LLMs achieve tremendous performance in many NLP applications, but their deployment on edge servers is difficult due to high requirement of computation, storage and power. In this paper, we used BERT as an example to show the possibility to save energy dissipation by employing approximate memories. Three compressed BERT models are implemented for the sentence emotion classification task with two weight formats (Float16 and INT8). Error injections with different BERs are applied to evaluate the effect of errors on the BERT models. Results show that errors on Float16 weights degrade the task performance dramatically, but the models with INT8 weights are almost unaffected by the errors even with BER of 10^{-2}. Therefore, aggressive scaling on the supply voltage of memories can be used for compressed BERT implementations with INT8 format. In the case of the memory considered in this paper this enables a reduction of up to 71.6% (46.9%) for the dynamic (static) power dissipation. These results prove the feasibility to save power/energy by using approximate memories that apply voltage scaling techniques on compressed and quantized LLMs.

REFERENCES

[1] OpenAI, Introducing ChatGPT, [Online]: https://openai.com/blog/chatgpt/, Nov. 2022.

[2] R. Yi, L. Guo, S. Wei, *et al.*, EdgeMoE: Fast On-Device Inference of MoE-based Large Language Models, arXiv:2309.16739, 2023.

[3] Z. Lin, G. Qu, Q. Chen, *et al.*, Pushing Large Language Models to the 6G Edge: Vision, Challenges, and Opportunities, arXiv:2309.16739, 2023.

[4] X. Zhu, J. Li, Y. Liu, *et al.*, A Survey on Model Compression for Large Language Models, arXiv:2308.07633v3, 2023.

[5] X. Ma, G. Fang and X. Wang, LLM-Pruner: On the Structural Pruning of Large Language Models, NeurIPS 2023.

[6] A. Radford, K. Narasimhan, T. Salimans, *et al.*, Improving Language Understanding by Generative Pre-Training, [Online]: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_ paper.pdf, 2018.

[7] T. B. Brown, B. Mann, N. Ryder, *et al.*, Language Models are Few-Shot Learners, NeurIPS 2020.

[8] OpenAI, GPT-4 Technical Report, arXiv:2303.08774, 2023.

[9] G W. Liu, F. Lombardi, and M. Shulte, "A retrospective and prospective view of approximate computing: point of view," Proceedings of the IEEE, vol. 108, no. 3, pp. 394-399, 2020.

[10] S. Kirolos and Y. Massoud, "Adaptive SRAM design for dynamic voltage scaling VLSI systems," in 2007 50th Midwest Symposium on Circuits and Systems, pp. 1297–1300, 2007.

[11] Bhati, M. Chang, Z. Chishti, S. Lu, and B. Jacob, "DRAM refresh mechanisms, penalties, and trade-offs," IEEE Transactions on Computers, vol. 65, no. 1, pp. 108–121, 2016.

[12] Y. Chen, X. Yang, F. Qiao, J. Han, Q. Wei, and H. Yang, "A multi-accuracy-level approximate memory architecture based on data significance analysis," in 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 385–390, 2016.

[13] S. Ataei and J. E. Stine, "A 64 kb approximate SRAM architecture for low-power video applications," IEEE Embedded Systems Letters, vol. 10, no. 1, pp. 10–13, 2018.

[14] P. Reviriego, S. Liu, Otmar Ertl and F. Lombardi, "Computing the Similarity Estimate Using Approximate Memory," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 3, pp. 1593-1604, 2022.

[15] D. T. Nguyen, N. H. Hung, H. Kim, and H. Lee, "An approximate memory architecture for energy saving in deep learning applications," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 67, no. 5, pp. 1588–1601, 2020.

[16] J. Devlin, M. Chang, K. Lee, et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL 2019, Minneapolis, Minnesota, USA.

[17] Z. Lan, M. Chen, S. Goodman, et al., "Albert: A lite bert for self-supervised learning of language representations," ICLR 2019.

[18] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," NeurIPS 2019.

[19] I. Turc, M. Chang, K. Lee, *et al.*, Well-read Students Learn Better: On the importance of Pre-trained Compact Models, arXiv:1908.08962v2, 2019.

[20] H. Wang, X. Kang and F. Ren, Emotion-Sentence-DistilBERT: A Sentence-BERT-Based Distillation Model for Text Emotion Classification, International Symposium on Artificial Intelligence and Robotics (ISAIR), Oct. 2022.

[21] Code for ALBERT , https://huggingface.co/bhadresh-savani/albert-base-v2-emotion.

[22] Code for DistilBERT, https://huggingface.co/esuriddick/distilbert-base-uncased-finetuned-emotion.

[23] Code for Tiny BERT, https://huggingface.co/gokuls/BERT-tiny-emotion-intent.

[24] F. Frustaci, M. Khayatzadeh, D. Blaauw, D. Sylvester, and M. Alioto, "SRAM for error-tolerant applications with dynamic energy-quality management in 28 nm CMOS," IEEE Journal of Solid-state circuits, vol. 50, no. 5, pp. 1310–1323, 2015.

[25] T. Mudger, "Power: A first-class architectural design constraint," Computer, vol. 34, no. 4, pp. 52-58 2001.