2024, 21(2 Special Issue): 0655-0661
https://doi.org/10.21123/bsj.2024.9743
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

The Effect Of Optimizers On The Generalizability Additive Neural Attention For Customer Support Twitter Dataset In Chatbot Application

Sinarwati Mohamad Suhaili*[1,2], ***Naomie Salim***[1], ***Mohamad Nazim Jambli***[3]

[1]Faculty of Computing, Universiti Teknologi Malaysia, 81310, Skudai, Johor, Malaysia.
[2]Centre of Pre-University, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia.
[3]Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Kota Samarahan, Sarawak, Malaysia.
*Corresponding Author.
PARS2023: Postgraduate Annual Research Seminars 2023.

Received 29/09/2023, Revised 10/02/2024, Accepted 12/02/2024, Published 25/02/2024

© 2022 The Author(s). Published by College of Science for Women, University of Baghdad.
This is an Open Access article distributed under the terms of the Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

When optimizing the performance of neural network-based chatbots, determining the optimizer is one of the most important aspects. Optimizers primarily control the adjustment of model parameters such as weight and bias to minimize a loss function during training. Adaptive optimizers such as ADAM have become a standard choice and are widely used for their invariant parameter updates' magnitudes concerning gradient scale variations, but often pose generalization problems. Alternatively, Stochastic Gradient Descent (SGD) with Momentum and the extension of ADAM, the ADAMW, offers several advantages. This study aims to compare and examine the effects of these optimizers on the chatbot CST dataset. The effectiveness of each optimizer is evaluated based on its sparse-categorical loss during training and BLEU in the inference phase, utilizing a neural generative attention-based additive scoring function. Despite memory constraints that limited ADAMW to ten epochs, this optimizer showed promising results compared to configurations using early stopping techniques. SGD provided higher BLEU scores for generalization but was very time-consuming. The results highlight the importance of finding a balance between optimization performance and computational efficiency, positioning ADAMW as a promising alternative when training efficiency and generalization are primary concerns.

Keywords: ADAM, ADAMW, Neural Network-based Chatbot, Optimizer, SGD.

Introduction

Integrating artificial intelligence (AI) through the use of neural networks is a widely used approach in various fields such as object and speech recognition, healthcare, and business, including chatbots. Chatbots based on neural networks typically aim to find the best function approximation by finding network parameters that minimize the error function during training data[1]. An error function measures how accurate the output of a model is compared to the actual output (target values). To improve the

2024, 21(2 Special Issue): 0655-0661
https://doi.org/10.21123/bsj.2024.9743
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

output (response), such parameters (weights) have to be optimized using optimization functions. Such parameters can be learned by training on labeled data (target values). Thus, the error is measured by comparing the values for each prediction y with the actual output (target values). The measurement of this error is associated with a loss or cost function[1]. To find an optimal weighting for the minimum loss function, the backpropagation algorithm can be used by adjusting the gradients of the loss function. Backpropagation is an algorithm for computing gradients from the output using the chain rule[1] and is an example of optimization techniques for training neural models based on gradients. However, the use of an algorithm based on finding gradients is very limited in its ability to find solutions for generalization. This limitation has led to the investigation of other optimization algorithms using decoupled decay regularization techniques such as ADAM and ADAMW, which are known for their superior performance. The efficiency of these chatbots in simulating human dialogues largely depends on the optimal tuning of the neural network weights, which is usually achieved by gradient-based algorithms such as backpropagation.

The optimizer determines how the network is updated based on the loss function. An optimizer concatenates the loss function and the model parameters by updating the model in response to the output of the loss function. Optimizers help minimize the loss function. There are two types of optimizers: gradient descent-based and adaptive optimizers. These different types of optimizers are based on an operational aspect where the learning rate is manually adjusted in the case of gradient descent algorithms such as batch gradient descent, stochastic gradient descent, and mini-batch gradient descent, while it is automatically adjusted in the case of adaptive algorithms, e.g., Adagrad, Adadelta RMSprop, ADAM, ADAMW, and ADAMAX, to name a few, as shown in Fig.1.

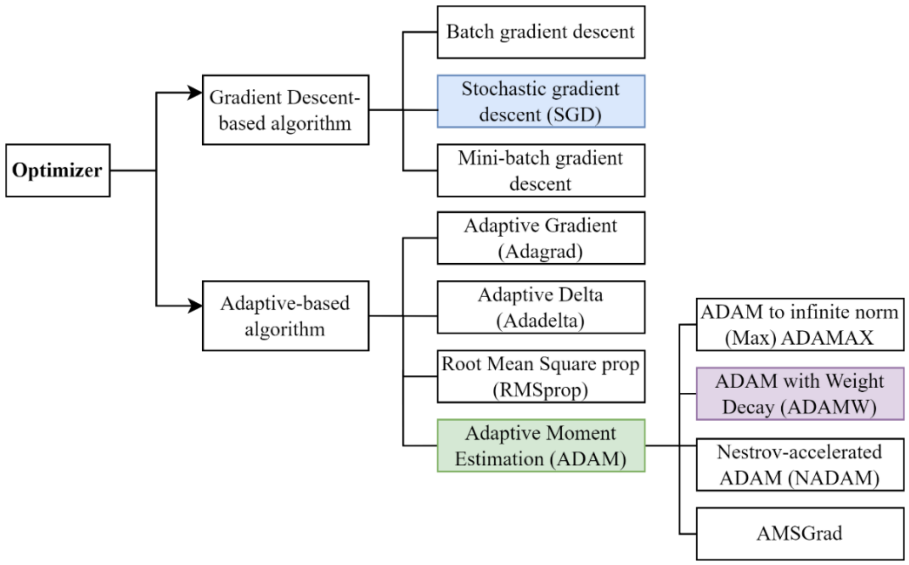

Figure 1. Optimizer Categorization.

Among commonly used optimizers, adaptive gradient-based methods such as ADAM have shown potential for performance improvements over SGD in some scenarios and have become the default choice in most studies[2,3]. However, recent studies show that ADAM, which is known for its scale-invariant parameter updates, is often criticized due to concerns about its generalization performance compared to SGD in image classification[2,4]. Although ADAMW — a variant in which the weight decay is managed after controlling the parameter-wise step size—presents an interesting alternative, there are few comparative studies between these optimizers. Therefore, this study aims to compare and investigate the effects of the optimizers SGD with Momentum, ADAM and ADAMW on the text chatbot CST dataset. The objective is to evaluate their performance based on

2024, 21(2 Special Issue): 0655-0661
https://doi.org/10.21123/bsj.2024.9743
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

training and validation losses and the BLEU scores for different search strategies to gain insight into the balance between optimization performance and computational efficiency. By revealing the performance nuances of these optimizers, this study seeks to guide the choice of optimization techniques in the development of neural network-based chatbots to improve their conversational quality and practicality.

The structure of this paper is outlined as follows: Section 2 presents the methodology of our experiments. The results obtained from the experiments are reported and discussed in Section 3, and finally, Section 4 summarizes the research findings and suggests directions for future studies.

Materials and Methods

This section provides an overview of the current methodological approach to research on the neural generative attention mechanism of the seq2seq model. The seq2seq learning task model is generally based on an encoder-decoder architecture consisting of three parts: encoder, context vector (final hidden/internal state vector), and decoder. To improve the performance of this structure, the augmentation layer of attention and the use of bi-LSTM are adopted in the encoder part. Before this model is performed, several preprocessing steps are required to conduct the current experimental study. The first step begins with splitting the initial dataset into a training set and a test set. The whole dataset is split into 75% and 35% for the training and validation/test sets, respectively. In this study, the publicly available dataset "Customer Support on a Twitter (CST)" from Kaggle was used to train and evaluate the models. The dataset should then be prepared for modeling. The preparation process includes preprocessing and feature extraction. For feature extraction, a transfer learning approach was adopted by using FastText pre-trained word embeddings to speed up training and increase model performance[5]. This approach considers knowledge transfer between networks trained on different datasets. The result of this step is incorporated into the neural generative attention model, which is trained with a training set. The training of this model to predict the response matches the ground-truth answers. The training process can be represented as minimizing the loss function $L(\theta)$, where θ represents the model parameters. The objective is to find the optimal θ that minimizes the difference between the predicted response and the ground truth, which can be mathematically defined by Eq. 1.

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{y}_i) \quad (1)$$

where $L(\theta)$ is the average loss over the training set, N is the number of examples in the training set, y_i refers as the ground truth for i, $\hat{y}_i$ is the predicted response for i generated by the model, and $L(y_i,\hat{y}_i)$ is the loss for i calculated using a loss function suitable for the problem at hand such as sparse-categorical cross-entropy loss for this case.

The optimization process to minimize $L(\theta)$ can be performed using a gradient-based optimizer such as SGD or adaptive methods like ADAM and ADAMW. These methods iteratively update parameter θ based on the gradient of the loss function with respect to θ [6]. These iterations continue until a stopping criterion is met, e.g., a predefined number of epochs or until the change in $L(\theta)$ falls below a certain threshold. The final result is an optimized set of parameters θ that can be used to make predictions that are very close to the ground truth. Finally, prepare the validation or test data set accordingly and use it to evaluate the models. Fig.2 illustrates the methodology used in this work.

This experiment is performed in a Python-dependent package on a deep neural network framework called TensorFlow[7] and Keras. The model was trained on a GPU with 3082 CUDA cores and a VRAM of 12 GB. The model was trained for 500 epochs (a high value since the study uses the early-stopping technique) and tested with a batch size of 64. The hidden size of the LSTM is tested with 480 units (the LSTM units that our memory space can hold). The three different optimizers (SGD, ADAM and ADAMW) were

2024, 21(2 Special Issue): 0655-0661
https://doi.org/10.21123/bsj.2024.9743
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

compared with a learning rate of 0.003 for the optimization[8]. The hyperparameter learning rate feeds into the optimization function. In the case of the SGD optimizer, only the momentum-accelerating gradient descent $\gamma \in \{0.09\}$ was tested. Here, 0 represents the vanilla gradient descent and 0.9 represents the convention[9]. A gradient clipping of 50.0 was also added to counteract the 'exploding gradient' problem. In this way, the gradients from growing exponentially and either overflowing (undefined values) or exceeding cliffs in the cost function. All weights and biases are initialized using the Xavier Uniform Glorot and Bengio (2010) distribution[10]. 300-dimensional pre-trained word embeddings for FastText were used. An early stopping technique with patience 5 was also employed to prevent overfitting. However, there are limitations to using the ADAMW optimizer since our memory resources are not occupied by the early stopping technique for training. Therefore, ten epochs for ADAMW were implemented without an early stopping technique for the model in this study. The hyperparameters and for training the models are listed in Table 1.

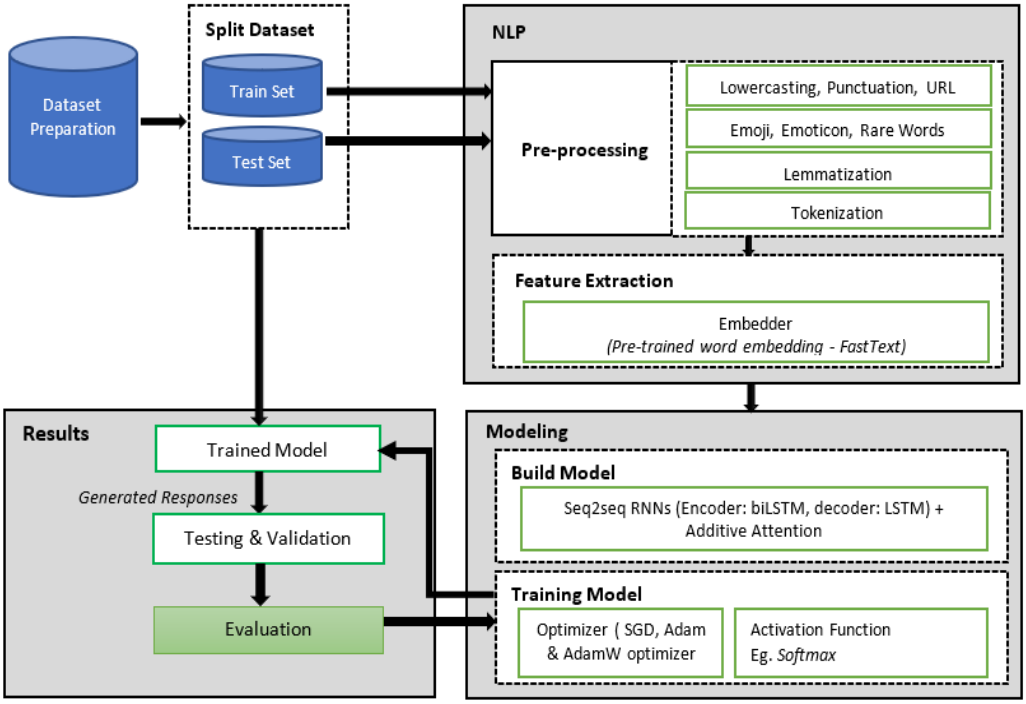

Figure 2. Illustration of the Methodology Step.

Table 1. Hyperparameter Setting.

Parameter	Config 1	Config 2	Config 3
Max Length Input	39	39	39
Embedding size	300	300	300
Batch Size	64	64	64
Hidden Unit	480	480	480
Learning rate	0.003	0.003	0.003
Clipvalue	0.5	0.5	0.5
Optimizer	Adam	AdamW	SGD
Learning_rate_decay	1.00E-06	none	none
Word embedding	FastText	FastText	FastText
Encoder type	Bidirectional	Bidirectional	Bidirectional

2024, 21(2 Special Issue): 0655-0661
https://doi.org/10.21123/bsj.2024.9743
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Results and Discussion

In this section, the experimental results of the model for the aforementioned dataset are presented. The experiment evaluated the performance of the different optimizers on the neural additive attention model with the pre-trained FastText embedding as an input feature to a model. Table 2 and Fig.3 show the performance results of the different optimizers on the model based on the sparse-categorical entropy loss during training and the BLEU scores metric in the inference phase. Due to memory issues, only ten epochs were run for Config 2, while an early stopping technique was used for the other configs during the training phase. The result shows that ADAM is the most effective optimizer during the training process, as it achieves the lowest training loss of 1.004115, which means that it converges the fastest during the training phase. On the other hand, SGD recorded the highest training loss of 1.557569, indicating a slower and less effective learning process. However, the validation loss result showed that ADAMW had the lowest validation loss of 1.138623, indicating that it is the most effective at generalizing and performing well on unseen data despite running on minimal 10 epochs. In addition, ADAMW achieved the highest BLEU score in the beam search scenario. This shows that ADAMW was able to learn efficiently in a minimal number of epochs. In the inference phase, the BLEU score analysis revealed nuances in the performance characteristics of the different optimizers. The highest BLEU in the greedy search was obtained by the SGD optimizer, indicating a better prediction of response quality with this search strategy. However, it is too time-consuming (almost a week to train a single model), which makes it seem less practical. This emphasizes the importance of considering multiple aspects when selecting an optimizer, including not only training efficiency but also generalization capabilities for unseen data and specific performance metrics under different inference techniques. Considering this aspect, the results highlight the importance of finding a balance between optimization performance and computational efficiency, positioning ADAMW as a promising alternative when training efficiency and generalization performance are primary concerns.

Table 2. Comparison of Different Optimizers based on the Neural Attention Model

	Training Phase		Inference Phase	
	FASTTEXT			
			BLEU	
Model	Loss	Val Loss	Greedy Search	Beam Search (k=3)
Config 1	1.004115	1.145985	0.433493	0.424751
Config 2	1.073800	1.138623	0.438506	0.440251
Config 3	1.557569	1.576885	0.440100	0.436540

2024, 21(2 Special Issue): 0655-0661
https://doi.org/10.21123/bsj.2024.9743
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

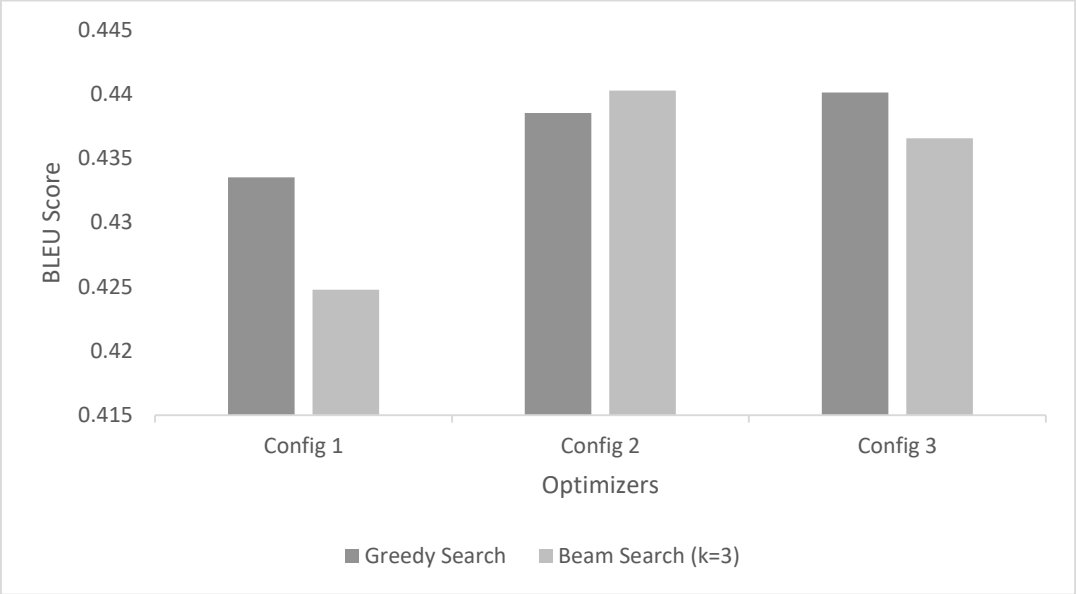

Figure 3. BLEU Score for Optimizers During the Inference Phase.

Conclusion

The study aimed to investigate the performance of different optimizers (SGD, ADAM, and ADAMW) on the neural additive attention model with FastText pre-trained embedding based on their sparse categorical loss during training and BLEU in the inference phase. During the training phase, ADAM proved to be the most efficient optimizer in minimizing loss. However, this did not directly translate into superior performance in all aspects of the inference phase, with ADAMW showing robust generalization and performing well on unseen data despite running on minimal 10 epochs, especially in beam search, while SGD was competitive in BLEU scores but very time-consuming. These results highlight the need to balance training efficiency with various aspects of validation and search strategies when selecting an optimizer. According to our result, ADAMW is a promising alternative when training efficiency and generalization performance are the main concerns as it can achieve comparable results for all evaluation aspects even though 10 epochs were used to train the model without implementing an early stopping technique due to memory constraints. If more epochs are used, it can be inferred that a satisfactory result can be obtained for the model performance.

Acknowledgment

The authors would like to express their gratitude to the Ministry of Higher Education Malaysia for partially funding this research under the Fundamental Research Grant Scheme (FRGS/1/2022/ICT06/UTM/01/1) with grant vote No. R.J130000.7851.5F568. Furthermore, appreciation is extended to Universiti Malaysia Sarawak (UNIMAS) and Universiti Teknologi Malaysia (UTM) for providing the necessary resources for this research work.

Authors' Declaration

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are ours. Furthermore, any Figures and images, that are not ours, have been included with the necessary permission for re-publication, which is attached to the manuscript.
- Ethical Clearance: The project was approved by the local ethical committee at the University of Universiti Teknologi Malaysia.

2024, 21(2 Special Issue): 0655-0661
https://doi.org/10.21123/bsj.2024.9743
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Authors' Contribution Statement

S. M.S. ,N. S. and M.N. J. contributed to the design and development of the research, to implement and analysis of the results, and to the manuscript.

References

1. Goodfellow I, Bengio Y, Courville A. Deep Learning. MIT Press; 2016. http://www.deeplearningbook.org.
2. Gupta M, Rajnish K, Bhattacharjee V. Impact of parameter tuning for optimizing deep neural network models for predicting software faults. Sci Program. 2021;1–17.https://doi.org/10.1155/2021/6662932.
3. Sulayman N. Deep Learning-based Predictive Model of mRNA Vaccine Deterioration: An Analysis of the Stanford COVID-19 mRNA Vaccine Dataset. Baghdad Sci. J . 2023;20(4(SI):1451-8. https://doi.org/10.21123/bsj.2023.8504.
4. Zhou P, Feng J, Ma C, et al. Towards theoretically understanding why sgd generalizes better than adam in deep learning. Adv Neural Inf Process Syst. 2020;33:21285–21296.
5. Wotaifi TA, Dhannoon BN. An Effective Hybrid Deep Neural Network for Arabic Fake News Detection. Baghdad Sci. J . 2023;20(4):1392. https://doi.org/10.21123/bsj.2023.7427.
6. Aggarwal CC. Neural networks and deep learning: A textbook. 2nd ed. Springer International Publishing; 2023. https://doi.org/10.1007/978-3-031-29642-0
7. Abadi M, Barham P, Chen J, et al. TensorFlow: A system for large-scale machine learning. 2016.
8. Mou L, Jin Z. Tree-Based Convolutional Neural Networks: Principles and Applications. 1st ed. Springer Publishing Company, Incorporated; 2018. https://doi.org/10.1007/978-981-13-1870-2
9. Tian Y, Zhang Y, Zhang H. Recent Advances in Stochastic Gradient Descent in Deep Learning. Mathematics. 2023;11(3):682. http://dx.doi.org/10.3390/math11030682.
10. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Teh YW, Titterington DM, editors. AISTATS, JMLR Proceedings, vol. 9; 2010. p. 249–256.

تأثير المُحسِّنات على قابلية التعميم للانتباه العصبي الإضافي لمجموعة بيانات تويتر لدعم العملاء في تطبيق Chatbot

سيناراواتي محمد سهيلي[1،2]، نعومي سليم[1]، محمد ناظم جمبلي[3]

[1]كلية الحاسبات، الجامعة التكنولوجية الماليزية، 81310، سكوداي، جوهور، ماليزيا.
[2]مركز ما قبل الجامعة، جامعة ماليزيا ساراواك، 94300 كوتا ساماراهان، ساراواك، ماليزيا.
[3]كلية علوم الكمبيوتر وتكنولوجيا المعلومات، جامعة ماليزيا ساراواك، كوتا ساماراهان، ساراواك، ماليزيا.

الخلاصة

عند تحسين أداء روبوتات الدردشة القائمة على الشبكة العصبية، يعد تحديد المحسن أحد أهم الجوانب. يتحكم المحسنون بشكل أساسي في تعديل معلمات النموذج مثل الوزن والتحيز لتقليل وظيفة الخسارة أثناء التدريب. أصبحت أدوات التحسين التكيفية مثل ADAM خيارًا قياسيًا وتستخدم على نطاق واسع لأحجام تحديثات المعلمات الثابتة الخاصة بها فيما يتعلق بتغيرات مقياس التدرج، ولكنها غالبًا ما تطرح مشاكل تعميم. وبدلاً من ذلك، يقدم مؤشر الهبوط التدرج العشوائي (SGD) مع الزخم وامتداد ADAM، ADAMW، العديد من المزايا تهدف هذه الدراسة إلى مقارنة وفحص تأثيرات هذه المُحسِّنات على مجموعة بيانات chatbot CST. يتم تقييم فعالية كل محسن بناءً على خسارته الفئوية المتفرقة أثناء التدريب وBLEU في مرحلة الاستدلال، وذلك باستخدام وظيفة تسجيل مضافة تعتمد على الاهتمام التوليدي العصبي. على الرغم من قيود الذاكرة التي حددت ADAMW بعشر فترات، أظهر هذا المحسن نتائج واعدة مقارنة بالتكوينات التي تستخدم تقنيات الإيقاف المبكر. قدمت SGD درجات BLEU أعلى للتعميم ولكنها كانت تستغرق وقتًا طويلاً للغاية. تسلط النتائج الضوء على أهمية إيجاد توازن بين أداء التحسين والكفاءة الحسابية، مما يضع ADAMW كبديل واعد عندما تكون كفاءة التدريب والتعميم هي الاهتمامات الأساسية

الكلمات المفتاحية: ADAM، ADAMW، Chatbot القائم على الشبكة العصبية، Optimizer، SGD.