www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

# Viewpoint On Energy-Aware Distributed Inference For Next-Word Prediction

[1]Mr. Ch Surya Prakash, [2] Sheik Anusha,

[1]Associate Professor,Department of MCA, Rajamahendri Institute of Engineering & Technology. Bhoopalapatnam, Near Pidimgoyyi,Rajahmundry,E.G.Dist.A.P. 533107.

[2]Student,Department of MCA, RajamahendriInstitute of Engineering & Technology. Bhoopalapatnam, Near Pidimgoyyi,Rajahmundry,E.G.Dist.A.P. 533107.

## Abstract—

In response to the immediate need for high-quality AI-generated contents (AIGC), natural language processing (NLP) services, particularly those offered at the edge (edge NLP), have developed. As an example of a popular edge NLP service for mobile keyboards on user devices, we examine distributed inference for next-word prediction in order to provide it more context. Keeping with this, we want to optimize related metrics, which means that we want to improve QoS by increasing the expected click-through rate (CTR), QoE by minimizing user impatience, and sustainability by keeping energy use within budget. We also consider the real world, where it is impossible to tell which natural language processing (NLP) model is the most accurate. Combining online learning and online control, we provide a novel distributed inference method for online next-word prediction with user impatience (DONUT) that balances the trade-offs among connected metrics and assesses the accuracy of model predictions. In accordance with our theoretical investigation, DONUT achieves sub-linear regret (loss of CTR), ensures restricted user impatience, and maintains budgetary energy use. We demonstrate DONUT's superior performance compared to competing baseline methodologies and its adaptability to various situations using numerical simulations. Index Terms—Online learning, online control, distributed inference, edge NLP, and next-word prediction.

## Introduction

A number of user-focused applications have begun using NLP services to produce high-quality AIGC (AI-generated content) during the AI revolution [1]. Robots can now understand human speech thanks to the combination of the two, which lowers the barrier to entry for AI and opens the door for a future when communicating with AI seems as natural as chatting to a person. For a long time, end customers seeking NLP services have relied on robust models housed on the cloud. Even while cloud services are quite accurate, they may have high latency and occasional unavailability.

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

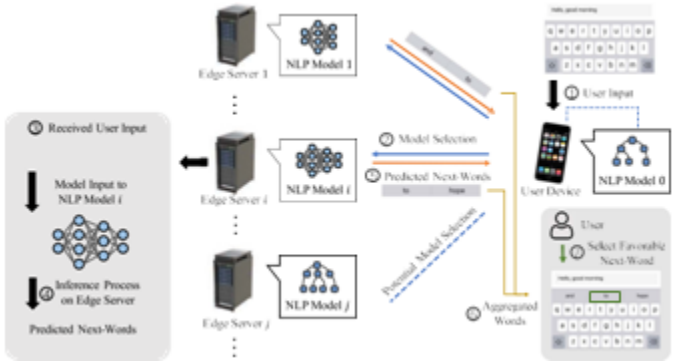



Figure 1: Edge Natural Language Processing for Next Word Prediction. 1) User input is generated at the user device. 2) A subset of NLP models is selected by the user. 3) The model input is received by the user. 4) Inference is performed on each model that has been selected. 5) The user device receives predicted next words from the selected NLP models. 6) The user device aggregates the words by removing duplicates and presents the aggregated words to the user. 7) The user selects one predicted word that they like.
via wireless networks, which makes maintaining real-time guarantees difficult [2]. One such alternative is edge NLP, which uses NLP services provided by servers located at the network's periphery. Thanks to the closeness of edge servers to consumer devices and their extensive computing capabilities, high-quality natural language processing (NLP) services with very quick response times are now accessible to end users [3, 4]. The majority of research on edge NLP has been on distributed training, with the primary issue being "How can we train NLP models across edge servers to enable NLP services?" [12] Here, we're taking a fresh approach by looking at the most effective methods to apply natural language processing models trained on edge networks, such as GPT [11], BERT [7], and N-gram [5]. This approach is known as distributed inference because it aims to use the inference capabilities of trained models across edge servers that are geographically distant. To set the scene, we examine distributed inference for next-word prediction, a popular and non-trivial consumer-facing NLP service (see Fig. 1). By anticipating user actions based on their mobile keyboard inputs, next-word prediction may streamline inputs [13].

TABLE I
COMPARISONOFINFERENCECAPABILITYAMONGVARIOUSNLPMODELSINTERMOFDIFFERENTFACTORS

| NLP Model Name | NLP Model Type | Prediction Accuracy | Inference Latency | Energy Consumption |
|---|---|---|---|---|
| N-gram [5] | Statistical Model | Low | Low | Low |
| ELMo [6] | Neural Model | Middle | High | High |
| BERT [7] | Neural Model | High | High | High |
| RoBERTa [8] | Neural Model | High | High | Middle |
| ALBERT [9] | Neural Model | High | High | Middle |
| DistilBERT [10] | Neural Model | Middle | Middle | Middle |
| GPT-3 [11] | Neural Model | High | High | High |

providing assistance. Ensemble learning provides strong support for this theory [14], since it is usual practice to use a mixture of experts (MOE) to improve performance over a single expert model [15]. The user device, however, is unable to communicate with an infinite number of NLP models due to rising overheads (such as energy cost and latency). This raises the question of which trained NLP models the user's device should utilize for distributed inference—a real-world model selection problem. The two reasons given below make this model selection challenge difficult and non-trivial. The inference capabilities of natural language processing (NLP) models is one of the criteria used to make a model selection choice [16]. In particular, we take into account three variables—prediction

accuracy, inference delay, and energy consumption—that might differ between NLP models. Although N gram often uses less energy and has shorter inference latency than transformer-based models[17], its prediction accuracy is typically poorer, as demonstrated in Table I. In addition, because deployed NLP models are not always customized to particular users, the prediction accuracy of these models is often user-dependent and unknown in advance. As a matter of fact, edge server NLP models include a diverse set of pre-trained models [18], the training corpus of which could differ significantly from the input corpus provided by users. The accuracy of the predictions may be estimated with the use of an online learning approach that gathers input via the dynamic selection of multiple models. Secondly, the sustainability of user devices, quality-of-service (QoS), and quality-of-experience (QoE) are all interrelated criteria in the model selection dilemma. The prediction click-through rate (CTR), or the rate at which the end user accepts the predicted words, is a standard metric for quality of service (QoS). It seems to reason that models with good accuracy in predictions should also have good prediction CTR and quality of service. In terms of quality of experience, we define it as the presence of user impatience owing to delayed predictions (i.e., when the time it takes to forecast the next word is longer than the maximum tolerance latency). Users may get very impatient and experience poor quality of experience due to models with long inference delays. For consumer devices with limited battery capacity, it is important to keep energy usage within budget in order to promote sustainability. As a result, models that use a lot of power might wind up going over their allotted amount and endangering sustainability. Adaptively adjusting the relative relevance of these connected measures requires an online control approach. In light of the above, it should be clear that the model selection issue is in essence a sequential decision-making problem with uncertainty, which is consistent with the context of brute-force learning [19]. Most bandit learning approaches, however, aren't suitable for immediate adoption since choosing a model requires a combination of online learning and online control in order to balance linked metrics and evaluate prediction accuracy. A possible consequence of inefficient online learning with erroneous estimate is that it may prejudice the selection of less-than-ideal NLP models for future control choices. However, it may be challenging to balance various goals, and poor feedback and learning efficiency might be the outcome of poorly implemented control choices. For example, the user device could misjudge the performance of some models if they are seldom used, leading to missed opportunities for optimum selections. Here, we present DONUT, a new eldistributed inference method for online next-word prediction with user impatience, as a solution to the crucial model selection challenge. What follows is a synopsis of our work's primary contributions and important outcomes: Creating models: Here, we delve into distributed inference's next-word prediction model selection challenge. Section III details our goal of maximizing prediction CTR for high QoS and minimizing user impatience for high QoE while keeping energy usage controlled. Our approach has the potential to be expanded to include other edge NLP services, such as emoji prediction, in addition to next-word prediction [20]. We also provide light on how to implement edge NLP in the future, which might be useful for services like Codex-based auto-programming tools [21] and ChatGPT-based dialog systems [11]. To address the model selection issue, we provide DONUT, an algorithm that combines online learning with online control, from the perspective of limited bandit learning (Section IV). We demonstrate in our theoretical analysis that DONUT minimizes the overall user impatience over time as $O(1/\alpha + V/\alpha)$, where T is the number of rounds and V and α are initial adjustable parameters, and the time-averaged regret as $O(1/V + \alpha/V + \log T/T)$. In addition, DONUT ensures that the user device's energy usage remains within the specified limits (Section V). To assess DONUT's efficacy, we run comprehensive simulations. They prove our theoretical analysis is correct and show that DONUT is superior than alternative web-based algorithms (Section VI)

## RELATED WORKS

Topic A: Natural Language Processing at the Edge The possibilities for delivering user-centric services via the use of edge-based resources have been greatly enhanced by the widespread adoption of edge computing. Edge NLP has arisen to capitalize on this trend by providing real-time, high-quality services like next-word prediction, emoji prediction, text categorization, audio recognition, etc. [22]. Here, we mostly go over papers that are relevant to the next-word prediction service. Training new natural language processing (NLP) models directly at the edge, using federated learning, or by fine-tuning pre-trained models, is the primary emphasis of most previous work [12]. A lightweight natural language processing model for next-word prediction termed CIFG (Coupled Input-Forget Gate) is trained via federated learning in work [13]. Additional work [23] uses a character-level recurrent neural network

based on the CIFG architecture to tackle the out-of-vocabulary issue. The training of customized models is addressed in work [24], which proposes three distinct algorithms—hypothesis-based clustering, data interpolation, and model interpolation—to address different real-world requirements. To train individualized models for next-word prediction, the authors of work [25] suggest combining sophisticated model aggression with an attention mechanism. For better next-word prediction performance, work[26] integrates federated fine-tuning with centralized model pre-training and pretrained word embeddings. Training high-performance distributed models lies at the heart of all the aforementioned federated learning studies. "How can we train new NLP models in a distributed manner for edge NLP?" is the question they want to answer. To help with edge NLP, our work takes a different tack. Our main objective is to take advantage of distributed inference, which allows trained models to make inferences at the edge. Consequently, "how to utilize trained NLP models in a distributed mannerforedgeNLP"[27] is where our attention is focused. Supporting next-word prediction services with high QoS and QoE in a sustainable way, distributed inference allows user devices to use heterogeneous trained NLP models at the edge. Part B: Edge Distributed Inference Distributed inference, which goes beyond edge NLP, is a new and exciting way to use edge-trained models to provide real-time applications with excellent quality. Much of the current literature on distributed inference focuses on how to divide up a single trained model across groups of untrained users. Inference overheads, including computational expenses and energy usage, are meant to be layer-wise distributed to edge servers or devices [28]. To take use of nearby computing resources for real-time inference, work[29] dynamically divides neural models across edge servers and devices. In order to reduce inference latency, research [30] separates the layers of directed acyclic graph (DAG) neural models into their own execution units. In order to optimize inference latency and throughput, researchers have been working on dividing up neural models and putting them into edge clusters [31]. Model partitioning applications have also started to surface. Research [32] investigates the use of partition-based distributed inference for real-time video analysis inside an Internet-of-Things system with limited resources. Partitioning models is done in work [33] to speed up distributed inference for edge video streams. Our study takes a novel tack by deploying several trained models at the edge rather than just one. Our goal is to improve the system's performance by using a mixture-of-expert method, which is achieved by modelselection and aggregate prediction. Thirdly, we formulate the system model and problems. Here we lay out the distributed inference system concept and problem formulation for next-word prediction. In particular, we zero in on the key models selection challenge, where we aim to maximize sustainability, quality of experience, and quality of service all at once. A non-trivial yet simple setup is the subject of this study. More specifically, we take into account an edge natural language processing system that offers the next-word prediction service to a single user, and it has $(N-1)$ edge servers. There is a singular trained natural language processing model installed on each user device and edge server. N models with different levels of prediction accuracy, inference delay, and energy consumption are thus implied. The models on the user device are indexed by zero, and we index them using the set $N \precsim \{0,1,...,N-1\}$. The system is assumed to function on a round-robin basis with a predetermined time horizon, denoted by $t \in \{0,...,T-1\}$. It is crucial to consider the end-to-end latency when accessing each model for inference [34], since edge NLP services are often very latency sensitive. In particular, the inference latency of the local model on the user device is what is meant by the end-to-end latency. The end-to-end latency for models running on an edge server is the sum of the inference latency and the transmission latency. The inference latency is the time it takes for the model to make predictions, while the transmission latency is the time it takes for the user device and the edge server to receive the results of those predictions. 2 Prior to making any model selection choices, we presume it is possible to quantify the associated inference latency and trans mission delay. Using analytical performance modeling with previous information about the NLP model [39] or a data-driven method using pre-trained machine learning models on different hardware configurations [37], [38] are two ways to evaluate the inference latency of NLP models [36]. Testing wireless channels using test packages [40], [41] or predicting latency using a model of the communication protocol of the channel (e.g., WiFi [42]) are two ways to assess the transmission delay of wireless communication between the user device and edge servers.

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

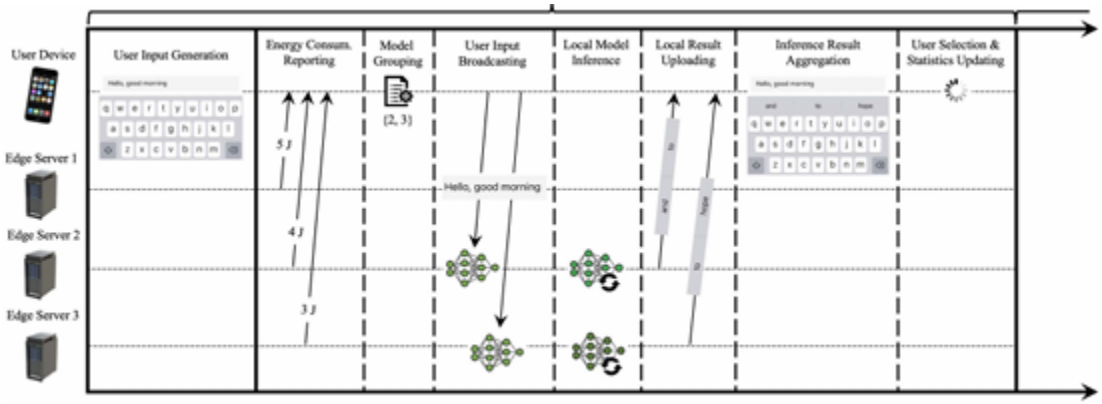



TABLE II SUMMARIZATION OF MAIN NOTATIONS

| Notation | Description |
|---|---|
| $T$ | Number of rounds for distributed inference for next-word prediction. |
| $n$ | Model selection number, *i.e.*, number of selected NLP models in each round. |
| $N$ | Total number of NLP models. |
| $\mathcal{N}$ | Set of NLP models on both the user device and edge servers with $\lvert\mathcal{N}\rvert \triangleq N$. |
| $S(t)$ | Set of selected NLP models in round $t$. |
| $a_i(t)$ | Click-through indicator for prediction results from NLP model $i$ in round $t$. |
| $s_i(t)$ | Indicator of in-time prediction for NLP model $i$ in round $t$. |
| $d_i(t)$ | Model selection decision on NLP model $i$ in round $t$. |
| $R_i(t)$ | Reward for selecting model $i$ for next-word prediction in round $t$. |
| $R(t)$ | Compound reward for the user device in round $t$. |
| $Z_i(t)$ | User impatience of NLP model $i$ in round $t$. |
| $W_i(t)$ | Energy consumed on the user device for next-word prediction with NLP model $i$ in round $t$. |
| $\mu_i$ | Prediction accuracy of NLP model $i$ with $\mu_i \triangleq \mathbb{E}[a_i(t)]$. |
| $p_i$ | Probability of in-time prediction for NLP model $i$ with $p_i \triangleq \mathbb{E}[s_i(t)]$. |
| $b$ | The energy budget on the user device. |

Experience Quality: User Restlessness To define QoE, we employ user impatience [44]. In natural language processing (NLP) models, user frustration builds up when there is a delay in either the selection process or the prediction, meaning that the total latency is longer than the user can bear. To be more precise, we may write Zi(t) as the related user impatience for each NLP model i in round t. User irritation with model i rises by one in each round if user does not pick model i or if model i's forecast is late; otherwise, it resets to one. From an intuitive standpoint, this reset ensures that the user's forecast demands are met promptly, alleviating any built-up irritation. Users' impatience with updates is therefore characterized as follows:

$$Z_i(t+1) = \begin{cases} Z_i(t) + 1, & s_i(t)d_i(t) = 0, \\ 1, & s_i(t)d_i(t) = 1. \end{cases} \tag{3}$$

, initial value is $Z_i(0) = 0$.

## THEORETICAL ANALYSIS

Here we provide the theoretical findings on energy usage, user frustration, and regret. To be more precise, we prove the virtual queue's stability in order to validate its fulfillment of the energy consumption requirement under DONUT. Following this, we provide the limits of total user impatience and time-averaged regret under DONUT,



Index in Cosmos
MAY 2025, Volume 15, ISSUE 2
UGC Approved Journal

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

respectively. Lastly, we demonstrate how user impatience and regret are affected by adjustable factors. The long-term energy consumption limitation in (6) can be met provided certain models and election policies are in place, which would make the proposed energy budget realistic. The maximum feasible region is defined as the collection of all possible energy budgets. For example, what if

TABLEIII IEEE TRANSACTIONS ONMOBILECOMPUTING, V (LEFT) AND ORDER OF α (RIGHT) VERSUS ORDER OF TIME AVERAGES

| Order of $V$ | Total User Impatience | Regret |
|---|---|---|
| $O(1/\sqrt{\log T})$ | $O(1/\sqrt{\log T})$ | $O(\sqrt{\log T})$ |
| $O(1)$ | $O(N)$ | $O(\sqrt{\log T/T})$ |
| $O(\sqrt{T/\log T})$ | $O(\sqrt{T/\log T})$ | $O(\sqrt{\log T/T})$ |

| Order of $\alpha$ | Total User Impatience | Regret |
|---|---|---|
| $O(1/V)$ | $O(V^2)$ | $O(1/V)$ |
| $O(1)$ | $O(V)$ | $O(1/V)$ |
| $O(V)$ | $O(1)$ | $O(1)$ |

ameters $V$ and $\alpha$, the virtual queue satisfies

$$\limsup_{T'\to\infty}\frac{1}{T'}\sum_{t=0}^{T'-1}\mathbb{E}\left[Q(t)\right]\leq\frac{C+N\alpha+nV}{\epsilon}<\infty,$$

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\sum_{i\in\mathcal{N}}Z_i(t)\right]\leq\left(\frac{N}{p_{\min}}+\frac{C+Vn}{\alpha p_{\min}}\right)\left(1+\frac{W_{\max}}{\epsilon}\right).$$

Sketch for the Proof of Theorems 1 and 2. Using Lyapunov optimization methods, we provide the evidence for queue stability and limits of user impatience. To start, we build a Lyapunov drift function to describe how the size of the queue backlog and user impatience change between rounds. The second step is to establish a maximum allowable value for the drift-plus-regret term by including the aim of regret minimization into the Lyapunov drift function; we then use the fact that the time-averaged queue backlog size is both limited and constrained to further constrain this value. Third, we build a supplementary strategy that prioritizes reducing user impatience by picking NLP models with the highest user impatience values right now. By using this auxiliary policy, we are able to determine the maximum allowable drift-plus-regret term for all model selection procedures. In the end, we use telescoping sum and conditional expectation to wrap up the evidence. Appendix B, which can be accessed online, contains the full proof. Notably, the fact that DONUT attains substantial queue stability for finite values of V and α indicates that constraint (6) is fulfilled, as shown by Theorem 1 [51]. Notably, the time-averaged total user impatience is constrained to be O(1/α + V/α), as shown by Theorem 2. The total user impatience is sub-linearly limited with the right order of V, for example, O(logT). Declaration No. 3 (Regret). The time-averaged respect across T rounds is constrained under DONUT.

## SIMULATION RESULTS

Ubuntu 20.04, Intel(R) Xeon(R) Gold 6230, and a 32 GB Quadro GV100 GPU are the specifications of the server that runs our simulations. We average each simulation outcome across 30 separate trials. One user device and fourteen edge servers, or fifteen NLP models, are the components of the system we're considering for next-word prediction using edge NLP. There are 105 operational rounds with the default value of T, and 300 and 1 for the parameters V and α, respectively. A DistilBERT [10] model is implemented on the user's device. The servers use three distinct natural language processing models: BERT (indexed from 1 to 8), RoBERTa [8] (indexed by 9 and 10), and ALBERT [9] (indexed from 11 to 14). The prediction accuracies of the Alltheabovemodels, which are pre-trained natural language processing models taken from the Transformers library [53], vary between 3 and 30%. Model selection on natural language processing (NLP) models uses 7–14 J/round of the user device's energy. If you choose to use distributed inference with each NLP model, they will all report the top k predictions (with k defaulting to 3). For natural language processing models, the in-time prediction probabilities are between 0.7 and 0.95.

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

Reuters-21578 [54], Customer Review Datasets [55], and Amazon Product Review Data [56] are the three corpora8 from which the user inputs are combined in each cycle. Table IV contains the default parameters for the simulation. Comparison of DONUT with Other Web-Based Algorithms Here we see how DONUT stacks up against some other popular internet algorithms. Focusing on distinct performance indicators and long-term limitations is where these algorithms diverge from DONUT. UCB1 prioritizes short-term prediction accuracy above longer-term restrictions [19]. For every i in N, the utility is defined as ˆ wi(t)=ˆ µi(t) according to UCB1 [19], where ˆ µi(t) is the UCB estimate given in (10). With UCB-Impatience, the user's impatience is minimized while prediction accuracy is maximized [45]. The utility, as defined under UCB-Impatience, is ˆ wi(t)=V ˆ µi(t)+αZi(t) for every i in N. Evaluation Using Various Criteria: Using a variety of indicators, we compare DONUT's performance to that of other online algorithms in Figure 3. Keep in mind that ensuring long-term limitations with minimal user impatience and energy consumption is just as important as measuring success by an individual indicator (e.g., prediction CTR). While DONUT's prediction CTR is lower than other online algorithms, it uses less energy and causes less user irritation. In comparison to UCB1 and UCB-Energy, DONUT reduces user frustration by up to 97% and energy usage by up to 16.6%. Under DONUT, as shown in Figure 3(b) and (c) respectively, the total user impatience over time is limited, and the energy consumption over time is maintained below the budget, which is shown as the black dash line in Figure 3(c)). The findings show that DONUTin successfully decreases user impatience while meeting the energy usage restriction. Comparison With the Use of an Integrated Metric: We provide an integrated measure to assess the overall performance of DONUT and other online algorithms in relation to prediction CTR, user impatience, and energy usage, in order to more clearly demonstrate its outperformances. Here is how the integrated metric 9 is defined:

$$\mathrm{IM} \triangleq \frac{e \cdot \text{Prediction CTR}}{\text{Energy Consumption}} + \frac{\text{Prediction CTR}}{\text{User Impatience}}, \qquad (19)$$

where the indication variable is located. To be more precise, e=1 if the energy consumption limitation belongs to (6). is fulfilled if and only if the time-averaged energy consumption is equal to or less than the energy that is budged, and it is set to zero otherwise. One would assume that an algorithm's performance will improve in direct correlation with its IM. As a result of user impatience and increased energy consumption per unit, a bigger IM leads to higher rises in prediction CTR. UCB-Energy ensures the energy usage limit and optimizes prediction CTR [57]. The utility ˆ wi(t) is defined as V ˆ µi(t) − αQ(t)Wi(t) for every i in N under UCB Energy. In terms of IM, impatience achieves worse results than DONUT. This finding provides further evidence that DONUT is capable of striking a good balance between prediction CTR, user impatience, and energy consumption.

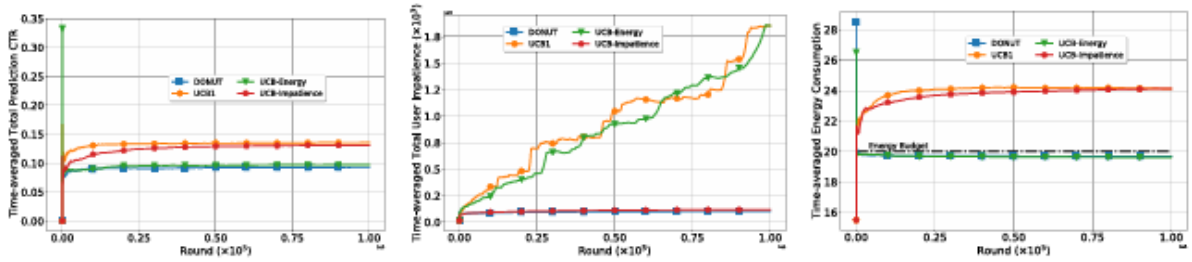



Fig. 3. Performances of DONUT and other online algorithms.

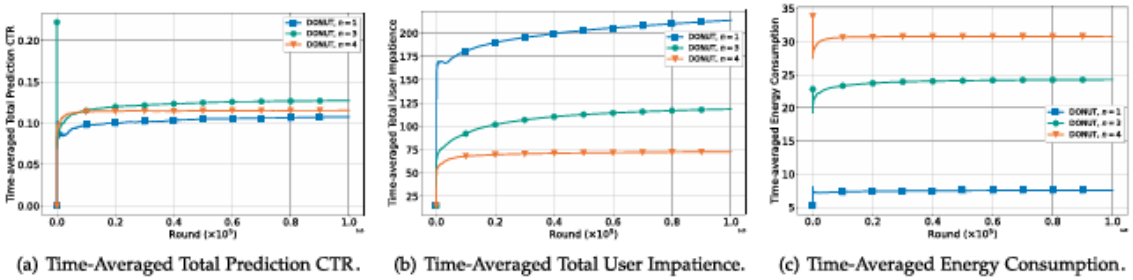
Parameters V and α and Their Impacts Two versions of DONUT are proposed to study the system performance under various learning techniques. In Algorithm 1, on line 4, the UCB term is replaced with alternative bandit learning algorithms. Donat-Epsilon acronym: The exploitation term ¯ µi(t), where i∈N, is substituted for the UCBestimatein(10) in each iteration. With a probability of 1 −, ∈{0,0.01,0.1}, the user device chooses the edge servers that have the greatest model utilities (18). In all other cases, it chooses n NLP models at random and in a uniform manner. "In each round, the exploration term γi(t) is substituted with the following expression: γi(t)= log



Index in Cosmos
MAY 2025, Volume 15, ISSUE 2
UGC Approved Journal

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

(max{T/(N •Ni(t)),1})/Ni(t)" [49]. Here, we assess how well DONUT and its variations perform with varying values of V and α. As shown in (18), V and α assess the readiness of selection for natural language processing models that exhibit low energy usage, high user impatience, and high prediction accuracies. Figure 5 displays the results of our evaluation of the impact of parameter V on the performances of DONUT and its variations. V may take on values between 1 and 500, but α is always set at 1. The chart shows that at increasing values of V, DONUT and its variants cause more user impatience and more energy usage, but less regrets. Take V as an example; from 1 to 500, DONUT experiences a 17.0% decrease in regrets, a 41.4% rise in user impatience, and a 4.2% increase in energy consumption. In particular, the importance of accurate predictions to the model's utility (18) climbs in direct proportion to the value of V. Consequently, DONUT would naturally prioritize increasing prediction CTR above decreasing user frustration and energy usage. Figure 5 shows that by adjusting V, DONUT and its variations are able to accomplish effective trade-offs among connected measures. In Fig. 6, we assess how parameter α impacts the performance of DONUT and its variables. V is set at 300 while the value of α varies from 1 to 200. According to the figure, DONT and its variations experience greater regrets as the value of α increases, while less user impatience and more energy consumption are the results. On average, DONUT and its variations produce 1.5% higher energy usage, 14.3% more regrets, and 7.7% less user irritation as the value of α changes from 1 to 200. These results demonstrate that DONUT and its derivatives accomplish adjustable trade-offs via parameter α. Interestingly, DONUT-Epsilon with =0 outperforms DONUT and other variables even when students do not engage in exploration while studying online. The reasoning behind this is that user impatiences will be minimized by selecting under-explored models when they reach a high enough level. To rephrase, our algorithm design indirectly encourages exploration via the online control mechanism. Part C: The Impact of Model Selection Quantities Optional Varieties for Fixed-Model Selection: We compare DONUT's performances under various model selection numbers in Figure 7. A value between 1 and 4 is assigned to the model selection number n, and 50 J is the energy budget b. Compared to a single option (n = 1), DONUT with several choices (n = 3, 4) has the potential to be superior in terms of prediction CTR, use impatience, and energy consumption, all while staying within budget. When it comes to prediction CTR, using several choices increases the likelihood of accurately predicting the user's next words and makes the prediction more trustworthy, even when dealing with unreliable wireless connections. In terms of user impatience, as seen in (3), more choices increase the likelihood of reducing irritation for numerous models.




(a) Time-Averaged Total Prediction CTR. (b) Time-Averaged Total User Impatience. (c) Time-Averaged Energy Consumption.

Fig. 7. Performances of DONUT under different model selection numbers.

energy use; nevertheless, DONUT ensures the long-term energy consumption limit, which maintains such usage under budget, even when frequent choices lead to increased consumption for wireless transmission. Quantities for Fixed and Flexible Model Selection: Figure 8 shows a comparison of DONUT's operating results with both rigid and dynamic model selection parameters. For situations where the model selection number is fixed, we assign 3 as the model selection number; for situations where the number is adjustable, we limit it to no more than 3. According to the findings, DONUT with a variable model selection number increases prediction CTR and decreases energy usage compared to the fixed model selection number instance, but it also increases user impatience. The rationale behind this is that DONUT focuses on a subset of NLP models due to its variable model selection number, which prevents it from picking NLP models with poor accuracy and high energy consumption. Therefore, it is unable to decrease user irritation by selecting models with high levels of impatience. Section D. Implications of Top-k Prediction Outcomes Fig. 9 shows our investigation into DONUT's performance in scenarios where NLP models provide top-k prediction results for k with varying values. We simulate a range of k values from 1 to 10, taking into account the usual range



Index in Cosmos
MAY 2025, Volume 15, ISSUE 2
UGC Approved Journal

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

of 5 to 25 anticipated next words shown to the user and the fixed value of 3 for the model selection number [13]. The values of V and α are 300 and 1, correspondingly. Table V summarizes the detailed prediction accuracies of all NLP models; Fig. 9(a) illustrates the evolution of prediction accuracies of NLP models indexed from 0 to 5 as k rises. Each NLP model's prediction accuracy grows monotonically with increasing values of k, according to the findings.

TABLE V PREDICTION ACCURACIES OF NLP MODELS WITH DIFFERENT VALUES OF k

| NLP Model Index | $k=1$ | $k=2$ | $k=3$ | $k=4$ | $k=5$ | $k=6$ | $k=7$ | $k=8$ | $k=9$ | $k=10$ |
|---|---|---|---|---|---|---|---|---|---|---|
| #0 | 2.78% | 4.26% | 5.16% | 6.04% | 6.85% | 7.48% | 8.02% | 8.67% | 9.09% | 9.64% |
| #1 | 5.38% | 7.44% | 8.99% | 10.00% | 11.12% | 11.96% | 12.70% | 13.54% | 14.26% | 14.84% |
| #2 | 5.68% | 7.93% | 9.45% | 10.50% | 11.35% | 12.26% | 13.00% | 13.89% | 14.53% | 15.18% |
| #3 | 4.70% | 6.67% | 7.88% | 8.83% | 9.70% | 10.54% | 11.27% | 11.84% | 12.36% | 13.01% |
| #4 | 5.12% | 7.14% | 8.56% | 9.54% | 10.47% | 11.10% | 11.89% | 12.59% | 13.47% | 14.06% |
| #5 | 4.58% | 6.67% | 8.13% | 9.32% | 10.25% | 11.26% | 12.06% | 12.78% | 13.49% | 14.15% |
| #6 | 4.74% | 7.49% | 9.37% | 10.98% | 12.15% | 13.40% | 14.27% | 15.27% | 15.97% | 16.80% |
| #7 | 1.44% | 2.54% | 3.39% | 4.21% | 4.87% | 5.49% | 5.90% | 6.38% | 7.04% | 7.53% |
| #8 | 5.31% | 7.50% | 9.30% | 10.72% | 11.70% | 12.48% | 13.35% | 14.19% | 14.91% | 15.45% |
| #9 | 11.76% | 16.20% | 19.06% | 21.27% | 23.16% | 24.92% | 26.15% | 27.32% | 28.41% | 29.28% |
| #10 | 13.58% | 18.40% | 21.57% | 24.16% | 26.19% | 28.19% | 29.73% | 31.13% | 32.08% | 33.20% |
| #11 | 5.77% | 8.39% | 10.38% | 11.77% | 13.16% | 14.41% | 15.46% | 16.31% | 17.03% | 17.70% |
| #12 | 6.72% | 9.62% | 11.53% | 13.21% | 14.46% | 15.57% | 16.64% | 17.62% | 18.58% | 19.44% |
| #13 | 3.99% | 5.86% | 7.52% | 8.78% | 9.93% | 10.78% | 11.49% | 12.22% | 12.71% | 13.31% |
| #14 | 6.79% | 9.62% | 11.33% | 12.55% | 13.60% | 14.60% | 15.42% | 16.28% | 16.86% | 17.48% |

Figures 9(b) and (c) show how DONUT performs with several values of k as V grows from 10 to 500. As an example, when V goes from 10 to 500, the overall prediction CTR increases by 20.3%, user impatience increases by 37.0%, and energy consumption increases by 4.10%, as shown by the blue curves in Fig. 9(b) and (c), respectively, for k = 3. A greater value of k for DONUT results in a higher overall prediction CTR, as shown in Figure 9(b), while obtaining the same amount of total user patience. For a total user impatience value of 90, for instance, the total prediction CTR grows by 158.13% from 1 to 9 for some value of k. The pattern in the change in total prediction CTR and total energy usage as k rises is comparable to what we see in Fig. 9(c). In particular, for a given total energy consumption number, DONUT produces better overall prediction results when k is larger. For a total energy consumption value of 19.6, for instance, the overall prediction CTR grows by 166.52% from 1 to 9 for all values of k. The rationale for this is because the number of authorized licensees, which are confined to the University of Punjab, increases as k becomes higher. Retrieved from IEEE Xplore on October 9, 2024 at 05:58:29 UTC. Limitations are in place. The WAN Getal Application. The user's device receives the results of 5707 predictions for next words: an analysis of energy-aware distributed inference. A greater prediction CTR is the outcome of the user's increased likelihood of discovering the favorable following words or phrases. In addition, when k becomes greater, the amount by which the overall prediction CTR is increased drops. As an example, there is a 73.77% rise in the overall prediction CTR when k grows from 1 to 3, but only a 10.76% increase when k increases from 7 to 9.

## Conclusion

With a focus on the standard next-word prediction service, we investigated distributed inference for edge NLP in this article. Our approach involves recasting the primary model selection issue as a multi-objective, online restricted optimization problem. Maximizing the predicted click-through rate, minimizing user impatience, and guaranteeing energy consumption within budget are all necessary to solve the issue. Our proposal, DONUT, seeks to address the model selection issue by combining online learning with online control, as seen from the perspective of limited bandit learning. Achieving sub-linear regret, ensuring constrained user impatience, and maintaining energy



Index in Cosmos
MAY 2025, Volume 15, ISSUE 2
UGC Approved Journal

consumption with budget are all shown by theoretical study of DONU. By an average of 160%, DONUT outperformed competing online algorithms according to the suggested integrated measure, as confirmed by the simulation results.

## REFERENCES

[1]. D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 2, pp. 604–624, Feb. 2021.
[2]. W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Internet Things J., vol. 3, no. 5, pp. 637–646,Oct. 2016.
[3]. D. Liu, H. Kong, X. Luo, W. Liu, and R. Subramaniam, "Bringing AI to edge: From deep learning's perspective," 2020, arXiv:2011.14808.
[4]. M.Chenetal.,"Distributedlearninginwirelessnetworks:Recentprogress and future challenges," IEEE J. Sel. Areas Commun., vol. 39, no. 12, pp. 3579–3605, Dec. 2021.
[5]. M. Chenet al., "Federated learning of n-gram language models," in Proc. 23rd Conf. Comput. Natural Lang. Learn., 2019, pp. 121–130.
[6]. M. E. Peters et al., "Deep contextualized word representations," in Proc. Annu.Conf.NorthAmer.ChapterAssoc.Comput.Linguistics:Hum.Lang. Technol., 2018, pp. 2227–2237.
[7]. [7] J. D.M.-W. C. Kenton and L. K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol., 2019, pp. 4171–4186.
[8]. Y.Liuet al., "RoBERTa: Arobustly optimized bert pretraining approach," 2019, arXiv:1907.11692.
[9]. Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite bert for self-supervised learning of language representations," 2019, arXiv:1909.11942.
[10]. V.Sanh,L.Debut,J.Chaumond,andT.Wolf,"DistilBERT,Adistilledversionofbert: Smaller, faster, cheaper andlighter," 2019,arXiv:1910.01108.
[11]. T. Brown et al., "Language models are few-shot learners," in Proc. 34th Int. Conf. Neural Inf. Process. Syst., 2020, Art. no. 159.
[12]. Z. Zhang, Y. Yang, Y. Dai, L. Qu, and Z. Xu, "When federated learning meets pre-trained language models' parameter-efficient tuning methods," 2022, arXiv:2212.10025.
[13]. A. Hard et al., "Federated learning for mobile keyboard prediction," 2018, arXiv:1811.03604.
[14]. X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," Front. Comput. Sci., vol. 14, no. 2, pp. 241–258, 2020. [15] Y. Zhou et al., "Mixture-of-experts with expert choice routing," in Proc. Int. Conf. Neural Inf. Process. Syst., 2022, pp. 7103–7114.
[15]. Q. Cao, Y. K. Lal, H. Trivedi, A. Balasubramanian, and N. Balasubramanian, "IrEne: Interpretable energy prediction for transformers," in Proc. Annu. Meeting Assoc. Comput. Linguistics, 2021, pp. 2145–2157.
[16]. A. Vaswani et al., "Attention is all you need," in Proc. 31st Int. Conf. Neural Inf. Process. Syst., 2017, pp. 6000–6010.
[17]. K. Liao, Y. Zhang, X. Ren, Q. Su, X. Sun, and B. He, "A global past future early exit method for accelerating inference of pre-trained language models," in Proc. North Amer. Chapter Assoc. Comput. Linguistics, 2021, pp. 2013–2023. [19] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," Mach. Learn., vol. 47, no. 2, pp. 235–256, 2002.
[18]. S.Ramaswamy,R.Mathews,K.Rao,andF.Beaufays,"Federatedlearning for emoji prediction in a mobile keyboard," 2019, arXiv:1906.04329.
[19]. M. Chen et al., "Evaluating large language models trained on code," 2021, arXiv:2107.03374.