# Offloading Adaptativo em Redes Neurais com Saídas Antecipadas: Equilíbrio e Compartilhamento de Recursos Através de Multi-Armed Bandits

**Ricardo S. Silva**[1], **Roberto G. Pacheco**[1], **Daniel S. Menasché**[2], **Heudson Mirandola**[2*]

[1]Universidade Federal Fluminense (UFF), Rio das Ostras – RJ – Brasil

[2]Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro – RJ – Brasil

{ri_soares, robertopacheco}@id.uff.br,
{sadoc@ic, mirandola@im}.ufrj.br

***Resumo.*** *Redes Neurais com Saídas Antecipadas (EENNs) reduzem custos de inferência ao classificar antecipadamente entradas em ramo lateral na borda, quando a confiança atinge um limiar fixo. Caso contrário, ocorre o offloading à nuvem. Contudo, um limiar fixo não se adapta às dinâmicas de aplicações reais e a competição de múltiplos dispositivos aos recursos da nuvem. Este trabalho modela a interação entre dispositivos via jogos estocásticos e multi-armed bandits para tornar o ajuste de limiares dinâmico, considerando restrições energéticas e disponibilidade de servidores. Os resultados numéricos mostram obtenção de equilíbrios de Nash aproximados.*

***Abstract.*** *Early-Exit Neural Networks (EENNs) reduce inference costs by enabling early classification at edge side branches when the confidence reaches a fixed threshold; otherwise, offloading to the cloud takes place. However, fixed thresholds fail to adapt to the dynamics of real-world applications and to the competition among multiple devices for cloud resources. This work models the interaction among devices through stochastic games and multi-armed bandits, enabling dynamic threshold adjustment while accounting for energy constraints and server availability. Numerical results demonstrate the emergence of approximate Nash equilibria.*

## 1. Introdução

A crescente complexidade das aplicações distribuídas tem reforçado a importância de sistemas de gerenciamento inteligentes para redes de computadores, especialmente diante da expansão da computação em borda [Satyanarayanan 2017]. Dispositivos de borda têm sido amplamente empregados em cenários como cidades inteligentes [Shi et al. 2016], veículos autônomos [Liu et al. 2020b], nos quais grandes volumes de dados sensoriais precisam ser processados com baixa latência e elevada confiabilidade. Nessas aplicações, a dependência exclusiva de infraestruturas de nuvem torna-se inviável devido aos atrasos de comunicação e à sobrecarga de rede, motivando a adoção de estratégias

*O trabalho contou com o apoio do CNPq (408255/2023-4, 444956/2024-7, 315106/2023-9); da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) – Código de Financiamento 001, e do projeto 315106/2023-9; da FAPERJ (E-26/204.268/2024) e da Finep PlatCiber.

híbridas que combinam processamento local e remoto por meio de mecanismos de *offloading* [Pacheco et al. 2021, Liu et al. 2020a].

Nesse contexto, a execução de inferência baseada em Redes Neurais Profundas (*Deep Neural Networks* – DNNs) impõe desafios significativos aos dispositivos móveis e de borda, uma vez que tais modelos demandam certo poder computacional e consumo energético [Mao et al. 2017]. Como alternativa, arquiteturas de DNNs com Saídas Antecipadas (*Early-exits Deep Neural Networks – EENNs*) [Teerapittayanon et al. 2016] permitem que decisões de classificação sejam tomadas em camadas intermediárias do modelo, reduzindo custos de inferência quando um limiar de confiança fixo e pré-definido é atingido. Caso contrário, o processamento restante é transferido à nuvem, possibilitando um equilíbrio entre acurácia, latência e uso de recursos [Rahmath P. et al. 2024]. Entretanto, a definição estática de limiar de confiança limita a eficiência dessas arquiteturas em ambientes dinâmicos, nos quais variações na carga da rede, na disponibilidade de servidores e nas restrições energéticas dos dispositivos influenciam diretamente o desempenho do sistema [Pacheco et al. 2025].

Além disso, cenários realistas de computação em borda envolvem múltiplos dispositivos competindo por recursos compartilhados de comunicação e processamento na infraestrutura de apoio. Essa concorrência cria interdependências entre as decisões individuais de *offloading*, de modo que a ação adotada por um dispositivo afeta a qualidade de serviço percebida pelos demais. Estratégias puramente locais tendem, portanto, a conduzir a soluções subótimas do ponto de vista global, evidenciando a necessidade de mecanismos de adaptação que considerem explicitamente a interação entre agentes. A modelagem dessas interações por meio de *teoria de jogos* fornece uma base formal para capturar tanto a natureza dinâmica do ambiente quanto os conflitos inerentes ao compartilhamento de recursos na borda [Zamzam et al. 2020].

O paradigma de aprendizado por reforço baseado em *Multi-Armed Bandits* (MAB) [Slivkins et al. 2019] apresenta-se como uma abordagem adequada para lidar com incertezas e variações temporais nesse tipo de cenário. Ao associar cada limiar de confiança a uma ação e modelar o retorno obtido em termos de métricas como latência, consumo energético e sucesso da inferência, torna-se possível ajustar dinamicamente o comportamento dos dispositivos com base na experiência acumulada. Embora técnicas baseadas em MAB já tenham sido exploradas em problemas de *offloading*, sua aplicação ao contexto de redes neurais com saídas antecipadas, considerando explicitamente múltiplos agentes concorrentes e restrições energéticas, ainda é pouco investigada.

Diante desse cenário, este trabalho propõe uma abordagem descentralizada para o ajuste dinâmico de limiares de confiança em DNNs com saídas antecipadas, modelando a interação entre dispositivos de borda como um jogo estocástico resolvido por meio de MABs. Conceitos da teoria dos jogos, como melhores respostas e equilíbrios, são empregados para analisar o comportamento coletivo dos dispositivos e explorar cenários de trade-off entre prioridade de acesso à infraestrutura de nuvem e capacidade energética disponível. A abordagem proposta busca promover decisões de *offloading* mais eficientes, adaptadas às condições do sistema e às limitações individuais dos dispositivos.

As principais contribuições deste trabalho podem ser resumidas da seguinte forma:

- a formulação do problema de *offloading* adaptativo em redes neurais com saídas

antecipadas como um jogo estocástico entre múltiplos dispositivos de borda;
- a proposição de um mecanismo baseado em MABs para o ajuste online de limiares de confiança, considerando simultaneamente restrições energéticas e disponibilidade da infraestrutura de apoio;
- a análise de diferentes cenários de concorrência, evidenciando os impactos das estratégias adotadas no desempenho global do sistema.

## 2. Trabalhos Relacionados

BranchyNet [Teerapittayanon et al. 2016] e SPINN [Laskaridis et al. 2020] são exemplos representativos de redes neurais profundas com saídas antecipadas, nas quais a decisão de interromper a inferência antes da camada final é fundamentada no grau de confiança da classificação. Na BranchyNet, essa decisão é tomada a partir da entropia da distribuição de probabilidades predita, permitindo a classificação antecipada sempre que a entropia seja inferior a um limiar fixo. De modo similar, a SPINN utiliza a probabilidade associada à classe mais provável como medida de confiança para determinar a ativação das saídas antecipadas. Outras arquiteturas, como SEE [Wang et al. 2019b], também adotam critérios baseados na confiança da predição para antecipar a classificação do modelo. Além dessas propostas, diversos trabalhos exploram o uso de DNNs com saídas antecipadas com o objetivo de reduzir o tempo de inferência e o consumo de recursos [Fang et al. 2020, Li et al. 2019]. Já o DynExit [Wang et al. 2019a] investiga a implementação de DNNs com saídas antecipadas em hardware FPGA, buscando otimizar simultaneamente latência e eficiência energética.

De modo geral, os trabalhos anteriores assumem limiares de confiança fixos ou definidos previamente [Fang et al. 2020, Li et al. 2019, Kim and Park 2020, Pacheco et al. 2021]. Em contraste, este trabalho propõe uma estratégia adaptativa que ajusta dinamicamente esses limiares por meio de aprendizado por reforço, permitindo uma resposta mais eficaz às mudanças no contexto do sistema, levando em conta o efeito da ação de um dispositivo na experiência dos demais.

Em contraste com os trabalhos discutidos anteriormente, LEE [Ju et al. 2021b], DEE [Ju et al. 2021a], UCBEE [Pacheco et al. 2024] e [Pacheco et al. 2025] formulam o problema de saídas antecipadas como um MAB, visando aprender políticas de decisão associadas ao processo de inferência. Embora compartilhe objetivos semelhantes, o presente trabalho distingue-se ao empregar MABs para ajustar dinamicamente os limiares que governam a ativação de saídas eficientes, enquanto LEE e DEE tratam explicitamente a seleção da saída ótima como a variável de controle do processo decisório.

Diferentemente de UCBEE [Pacheco et al. 2024] e [Pacheco et al. 2025], esta proposta considera um cenário com múltiplos agentes, nos quais há interação e competição pelo acesso aos recursos da nuvem. Tal dinâmica é modelada por meio de teoria dos jogos, ao passo que os trabalhos anteriores assumem um único dispositivo de borda operando de forma isolada, com acesso irrestrito aos recursos da nuvem e ausência de competição.

## 3. Redes Neurais com Saídas Antecipadas

As Redes Neurais Profundas com Saídas Antecipadas (*Early-exit* Deep Neural Networks – EENNs) incorporam ramos laterais (saídas antecipadas) em modelos de DNNs tradicionais, como o MobileNetV2 [Krizhevsky et al. 2012]. Esses ramos laterais permitem

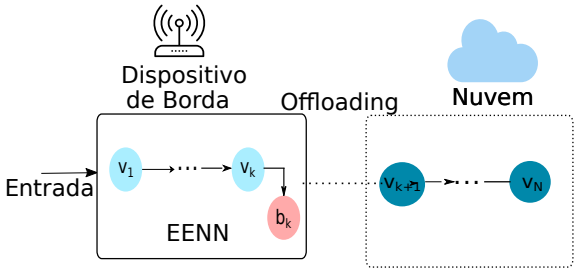


**Figura 1. Ilustração do cenário de *offloading* adaptativo usando EENN.**

que as imagens de entrada sejam classificadas antecipadamente ainda nas camadas intermediárias, caso a classificação seja suficientemente confiante. A Figura 1 ilustra um modelo de EENN com um ramo lateral utilizado para implementar um cenário de inferência colaborativa entre borda e nuvem.

Nesse cenário, após o dispositivo de borda receber uma imagem de entrada $\boldsymbol{x}$, o modelo de EENN a processa camada por camada até atingir o ramo lateral que possui um classificador intermediário, no qual é obtido o vetor de saída intermediário $\boldsymbol{z}_I$. Em seguida, o modelo aplica a função *softmax* para gerar o vetor de probabilidade $\boldsymbol{p}_I = \text{softmax}(\boldsymbol{z}_I)$, em que $\text{softmax}(\boldsymbol{z}_I) \propto \exp(\boldsymbol{z}_I)$. Cada componente de $\boldsymbol{p}_I$ representa a probabilidade de $\boldsymbol{x}$ pertencer a uma classe específica. A partir do vetor $\boldsymbol{p}_I$, é possível calcular a confiança intermediária de classificação da entrada $\boldsymbol{x}$ como $C_I = \max \boldsymbol{p}_I$. Se a confiança $C_I \geq \alpha$, em que $\alpha$ é o limiar de confiança, o ramo lateral classifica $\boldsymbol{x}$ como $\hat{y}_I = \text{argmax}(\boldsymbol{p}_I)$, encerrando o processo de inferência no próprio dispositivo de borda, sem a necessidade de realizar o *offloading* para a nuvem. Caso contrário, se $C_I < \alpha$, o dispositivo de borda transfere os dados para a nuvem, que processa as camadas subsequentes até a camada final, gerando o vetor final de probabilidade, o que permite obter a confiança final $C_L = \max \boldsymbol{p}_L$ e a classificação $\hat{y}_L = \text{argmax}(\boldsymbol{p}_L)$.

O processamento adicional decorrente do envio de dados à nuvem implica um custo adicional, que pode ser modelado como uma sobrecarga $o$, cuja magnitude varia conforme o contexto da aplicação. No cenário em questão, a sobrecarga abrange tanto o atraso de comunicação necessário para enviar os dados do dispositivo de borda para a nuvem quanto o tempo de inferência das camadas remanescentes executadas no ambiente remoto. Por outro lado, em aplicações estritamente móveis, nas quais todo o processamento ocorre localmente, a sobrecarga está relacionada ao tempo de execução das camadas posteriores no próprio dispositivo, bem como, potencialmente, ao consumo energético associado. Neste trabalho, levamos em conta o fato de que ao gerar sobrecarga na rede, um usuário pode afetar os demais, caracterizando assim um jogo estocástico.

Embora arquiteturas EENN possam incorporar múltiplos ramos de saída antecipada, a análise desenvolvida considera um único ramo lateral, a fim de reduzir a complexidade do modelo.

## 4. Aprendizado por Meio de Multi-Armed Bandits

Tradicionalmente, os modelos de EENNs utilizam um limiar de confiança fixo $\alpha$ para decidir se uma dada imagem de entrada deve ser classificada antecipadamente ou deve seguir sendo processada pelas camadas posteriores. Contudo, aplicações reais podem necessitar de um limiar adaptativo para se adaptar ao contexto. Com objetivo de selecionar

dinamicamente o limiar de confiança ótimo $\alpha^*$ de acordo com o contexto, o problema é modelado como um problema de aprendizado por reforço baseado em Multi-Armed Bandits (MAB), aplicado às decisões de saídas antecipadas. A seguir nesta seção, apresenta-se o procedimento (detalhado no **Algoritmo 1**) em detalhes.

A cada rodada $t$, uma imagem de entrada $\boldsymbol{x}_t$ é recebida pelo dispositivo de borda $j$, que executa um algoritmo para escolher um limiar de confiança $\alpha_{j,t} \in \mathcal{A}$, onde $\mathcal{A}$ denota o conjunto de possíveis limiares disponíveis para serem escolhidos. A cada seleção de uma ação $\alpha_{j,t}$, associa-se uma recompensa instantânea $r(\alpha_{j,t})$, a qual reflete o impacto da decisão de offloading. Ao receber a imagem de entrada $\boldsymbol{x}_t$, a confiança intermediária $C_{I,j}(\boldsymbol{x}_t)$ e a confiança final $C_{I,j}(\boldsymbol{x}_t)$ obtidas pelo dispositivo de borda $j$ são inicialmente desconhecidas até que a imagem seja processada pelo modelo de EENN.

Posteriormente, conforme descrito na Seção 3, a confiança obtida na camada intermediária do dispositivo de borda $j$, denotada por $C_{I,j}(\boldsymbol{x}_t)$, é comparada ao limiar de confiança selecionado $\alpha_{j,t}$. Quando $C_{I,j}(\boldsymbol{x}_t) \geq \alpha_{j,t}$, a inferência é finalizada localmente no próprio dispositivo de borda, e a amostra $\boldsymbol{x}_t$ é classificada no ramo lateral do modelo. Nessa situação, não há custo adicional de comunicação nem ganho de confiança decorrente do processamento em nuvem, i.e., a recompensa instantânea nula: $r_j(\alpha_{j,t}) = 0$.

Por outro lado, quando $C_{I,j}(\boldsymbol{x}_t) < \alpha_{j,t}$, o dispositivo de borda $j$ realiza o *offloading* da amostra $\boldsymbol{x}_t$ à nuvem, onde a inferência é concluída na camada final do modelo. Nesse caso, a recompensa instantânea é composta por dois termos. O ganho de confiança $\Delta C_j(\boldsymbol{x}_t)$ obtido ao avançar da camada intermediária para a camada final, definido por $\Delta C_j(\boldsymbol{x}_t) = \max\big(C_{L,j}(\boldsymbol{x}_t) - C_{I,j}(\boldsymbol{x}_t),\, 0\big)$, atua como um *proxy de ganho de acurácia*, assumindo valores elevados quando a camada final do modelo apresenta uma confiança significativamente superior à obtida no ramo lateral, e valor nulo quando a classificação realizada na saída antecipada já possui nível de confiança equivalente ao da camada final da EENN [Bajpai and Hanawal 2025, Casale and Roveri 2023, Pacheco et al. 2024]. O outro termo $\eta_j(o_{j,t})$ está associado ao custo energético da comunicação, modelado por uma função estritamente decrescente em relação ao número de amostras $o_{j,t}$ já transferidas pelo dispositivo, refletindo o impacto acumulado do consumo energético associado ao *offloading*. Este trabalho assume $\eta_j(o_{j,t}) \in [0, 1]$, de modo que $\Delta C_j$ e $\eta_j(o_{j,t})$ sejam grandezas comensuráveis. Portanto, quando ocorre o *offloading* bem sucedido, a recompensa instantânea é dada por $r_j(\alpha_{j,t}) = \Delta C_j(\boldsymbol{x}_t) - \eta_j(o_{j,t})$. Em resumo, a cada escolha de limiar realizada pelo dispositivo de borda, há uma recompensa instantânea associada da seguinte forma:

$$r_j(\alpha_{jt}) = \begin{cases} 0, & \text{se } C_{I,j}(\boldsymbol{x}_t) \geq \alpha_{jt} \\ -\eta_j(o_{j,t}), & \text{se } C_{I,j}(\boldsymbol{x}_t) < \alpha_{jt} \text{ e } \mathscr{T}_{j,t} = 0 \\ \Delta C_j(\boldsymbol{x}_t) - \eta_j(o_{j,t}), & \text{caso contrário,} \end{cases} \tag{1}$$

onde $\mathscr{T}_{j,t} \in \{0, 1\}$ denota uma variável aleatória para modelar o sucesso do usuário $j$ ao realizar um *offloading*, de modo que $\mathscr{T}_{j,t} = 0$ indica que houve um fracasso, e.g., em razão da disputa por recursos entre demais agentes e $\mathscr{T}_{j,t} = 1$ denota um sucesso no acesso à nuvem.[1]

[1]Por restrições de espaço, a caracterização detalhada de $\mathscr{T}_{j,t}$ é apresentada em `https://github.com/selsoaress/SBRC-2026-Offloading-Adaptativo`.

Baseado na recompensa instantânea $r_j(\alpha_{j,t})$, define-se a recompensa média associada à escolha do limiar $\alpha_{j,t}$ como

$$\mathbb{E}[r_j(\alpha_{j,t})] = \mathbb{E}[\Delta C_j(\boldsymbol{x}_t) - \eta_j(o_{j,t}) \mid C_{I,j}(\boldsymbol{x}_t) < \alpha_{j,t}] \; \mathbb{P}[C_{I,j}(\boldsymbol{x}_t) < \alpha_{j,t}] \,, \qquad (2)$$

onde assumimos que $\Delta C_j(\boldsymbol{x}_t) = 0$ se $\mathscr{T}_{j,t} = 0$, de tal forma que a equação acima segue de (1). O objetivo deste trabalho consiste em determinar o limiar de confiança ótimo $\alpha^*$ que maximiza a recompensa média esperada, definido por $\alpha^* = \arg\max_{\alpha \in \mathcal{A}} \mathbb{E}[r(\alpha_{j,t})]$, onde onde $\mathcal{A}$ denota o conjunto de limiares de confiança disponíveis. Uma política $\Pi : \mathcal{H}_t \to \mathcal{A}$ é definida como uma regra de decisão que, a cada rodada $t$, mapeia o histórico de interações observadas até o instante $t-1$, denotado por

$$\mathcal{H}_t = (\alpha_1, r(\alpha_1)), \ldots, (\alpha_{t-1}, r(\alpha_{t-1})) \qquad (3)$$

até $t-1$ para a seleção de um limiar de confiança $\alpha_t \in \mathcal{A}$, isto é, $\alpha_t = \Pi(\mathcal{H}_t)$. Para avaliar o desempenho de uma política $\Pi$ em $T \in \mathbb{N}$ rodadas, define-se uma métrica denominada arrependimento esperado (*expected regret*) dada por:

$$R(\Pi, T) = \sum_{t=1}^{T} \big(r_j(\alpha_j^*) - r_j(\alpha_{jt})\big) = T \cdot r_j(\alpha_j^*) - \sum_{t=1}^{T} r_j(\alpha_{j,t}) \qquad (4)$$

O pseudo-código do algoritmo baseado em UCB para a seleção adaptativa dos limiares de confiança é apresentado no **Algoritmo 1**. Como parâmetros de entrada, o algoritmo recebe o coeficiente $\tilde{c}$, responsável por regular o equilíbrio entre exploração e aproveitamento, e o número total de limiares disponíveis, denotado por $K$.

Na etapa inicial, cada um dos $K$ limiares é empregado exatamente uma vez, de modo a garantir uma observação inicial de recompensa para todas as ações. A partir das rodadas seguintes, o limiar a ser utilizado é escolhido com base no maior valor do índice UCB calculado a partir das estatísticas acumuladas. Uma vez selecionado o limiar, a confiança obtida na camada intermediária é comparada ao valor adotado. Caso essa confiança seja superior ao limiar, a inferência é encerrada antecipadamente no ramo lateral da rede. Caso contrário, o processamento prossegue até as camadas finais do modelo, sendo considerada como saída a classificação associada ao maior nível de confiança entre as camadas avaliadas.

Após a obtenção da decisão final, as estatísticas do algoritmo são atualizadas. Em particular, $N_t(\alpha_t)$ corresponde ao número de vezes em que o limiar $\alpha_t$ foi selecionado até a rodada $t$, enquanto $Q_t(\alpha_t)$ representa a recompensa média estimada associada a esse limiar. Essas informações são então utilizadas para orientar as decisões nas rodadas subsequentes.

## 5. Jogos Estocásticos

A modelagem baseada em jogos estocásticos fornece uma estrutura sistemática para a análise das interações entre múltiplos agentes em ambientes dinâmicos e concorrentes [Albrecht et al. 2024]. A estocasticidade decorre da incerteza inerente às condições da rede, à competição entre dispositivos e ao compartilhamento de recursos na ausência de um mecanismo de orquestração centralizado. Cada agente possui informações locais

**Algorithm 1:** MAB (UCB) no dispositivo $j$

**Entrada:** conjunto de limiares $\mathcal{A} = \{\alpha^{(1)}, \ldots, \alpha^{(K)}\}$; parâmetro $\tilde{c} > 0$; horizonte $T$
**Início:** Para cada $k = 1, \ldots, K$: $Q_{j,k} \leftarrow 0$ e $N_{j,k} \leftarrow 0$

**for** $t = 1, \ldots, T$ **do**
  Receber amostra $\boldsymbol{x}_t$
  **if** $t \leq K$ **then**
    $k_t \leftarrow t$ `// exploração inicial`
  **else**
    $k_t \leftarrow \underset{k \in \{1,\ldots,K\}}{\arg\max} \left( Q_{j,k} + \tilde{c}\sqrt{\frac{2 \ln t}{N_{j,k}}} \right)$
  **end**
  $\alpha_{j,t} \leftarrow \alpha^{(k_t)}$
  Calcular $C_{I,j}(\boldsymbol{x}_t)$
  **if** $C_{I,j}(\boldsymbol{x}_t) \geq \alpha_{j,t}$ **then**
    $r_{j,t} \leftarrow 0$ `// inferência encerrada na borda`
  **else**
    Tentar *offloading* e observar $\mathscr{T}_{j,t} \in \{0, 1\}$
    **if** $\mathscr{T}_{j,t} = 0$ **then**
      $r_{j,t} \leftarrow -\eta_j(o_{j,t})$
    **else**
      Calcular $C_{L,j}(\boldsymbol{x}_t)$ e $\Delta C_j(\boldsymbol{x}_t) \leftarrow \max\big(C_{L,j}(\boldsymbol{x}_t) - C_{I,j}(\boldsymbol{x}_t), 0\big)$
      $r_{j,t} \leftarrow \Delta C_j(\boldsymbol{x}_t) - \eta_j(o_{j,t})$
    **end**
  **end**
  $N_{j,k_t} \leftarrow N_{j,k_t} + 1$
  $Q_{j,k_t} \leftarrow Q_{j,k_t} + \frac{1}{N_{j,k_t}}\big(r_{j,t} - Q_{j,k_t}\big)$
**end**

sobre o sistema, consequência da descentralização do mesmo. Nesse cenário, cada agente busca otimizar seu próprio desempenho com base em informações reduzidas, tornando desejável a abstração dos estados observáveis ao conjunto mínimo necessário para a tomada de decisão.

### 5.1. Fundamentos

O jogo estocástico é definido pela tripla $(\mathscr{J}, \mathscr{A}, \mathscr{R})$ que compreende um conjunto de agentes $\mathscr{J} = \{1, \cdots, n\}$, um espaço de ações conjuntas $\mathscr{A}$ e uma função de recompensa $\mathscr{R}_j : \mathscr{A} \to \mathbb{R}$ para cada integrante. No modelo proposto, cada agente $j \in \mathscr{J}$ corresponde a um dispositivo de borda que possui um conjunto finito de ações possíveis $\mathscr{A}_j$. No contexto analisado, essas ações correspondem às decisões operacionais, que são *offload*, isto é, processar a inferência parcialmente na nuvem, ou *borda*, isto é, realizar a inferência integralmente no dispositivo de borda. Portanto, o conjunto $\mathscr{A}_j$ de cada agente é dado como $\mathscr{A}_j = \{\textbf{offload}, \textbf{borda}\}$, representando a decisão de realizar ou não o *offloading*. Adicionalmente, define-se uma função de recompensa individual $\mathscr{R}_j : \mathscr{A} \to \mathbb{R}$ associada a cada combinação de ações dos agentes que quantifica o desempenho percebido por cada dispositivo em função das decisões conjuntas tomadas no sistema.[2]

[2]Note que no jogo estocástico, a ação é offload ou borda, mas os jogadores tomam a ação de forma mista. A escolha do threshold, conforme Algoritmo 1, é que define a probabilidade de adotar cada ação.

Nos interessam, particularmente, as múltiplas iterações de jogos normais, de modo que cada turno $t$ corresponda a um jogo em forma normal. Com esta base estabelecida, podemos analisar com maior precisão o impacto do aprendizado agregado entre agentes, conforme discutido a seguir.

### 5.2. Definição do Jogo Estocástico EENN

Uma vez estabelecido o modelo básico de jogo que sustenta a análise, é possível introduzir complexidade ao incorporar estados. Consideraremos o estado como o número de transmissões ao servidor; desse modo, poderemos interpretar o desempenho dos dispositivos de acordo com uma métrica de pontuação. Denotamos por $o_{i,t}$ o número acumulado de offloadings realizados pelo dispositivo $i$ até o turno $t$. Assim, o estado global no instante $t$ pode ser representado de forma compacta por $s_t = (o_{1,t}, o_{2,t}, \ldots, o_{n,t})$. A cada turno $t$, o sistema transita de $s_t$ para um novo estado $s_{t+1}$ segundo uma probabilidade de transição $\mathcal{T}(s_{t+1} \mid s_t, A_t)$, em que $A_t = (A_{1,t}, A_{2,t}, \ldots, A_{n,t})$ é o vetor de ações realizadas no tempo $t$. A probabilidade $\mathcal{T}$ recebe esta notação por tratar-se de uma probabilidade de transição entre estados.

Do ponto de vista de cada dispositivo de borda, o sistema comporta-se como um processo de decisão de Markov (*Markov Decision Process*) [Altman 1999, Albrecht et al. 2024], no qual cada aparelho armazena o próprio estado (e.g., $o_{i,t}$) e ignora o dos demais. Esse acesso a informação incompleta é o que caracteriza o modelo como um jogo de observação parcial.

Ao retomar as limitações energéticas do hardware, o espaço de estados possíveis é delimitado por uma série de restrições sobre o consumo esperado, ou seja:

$$\eta_{i,t}(o_{i,t}) \leq \eta_{i,\max}, \quad \forall i \in \mathcal{J}. \tag{6}$$

Estados que representam o consumo total ou energia insuficiente para o *offloading* em todo o conjunto de dispositivos são denominados *estados terminais*. O jogo manifesta-se por meio da transição entre o conjunto de estados possíveis. Ao longo do processo de aprendizado, exploração e adaptação dos agentes, surgem equilíbrios ou estabilidades que podem ser explorados para compreender os impactos da heterogeneidade dos dispositivos em cenários de compartilhamento de recursos.

O recurso principal considerado neste trabalho é a energia disponível nos dispositivos de borda. Assim, toda vez que um dispositivo realiza *offloading*, ocorre consumo energético sem qualquer mecanismo de renovação considerado no horizonte de análise. Como consequência, em execuções suficientemente longas, a energia de todos os dispositivos tende a se exaurir, levando o sistema a um estado em que apenas classificações locais são possíveis e a infraestrutura de nuvem permanece ociosa. A modelagem de processos de recarga ou colheita de energia, bem como seus impactos na dinâmica do jogo, é considerada fora do escopo deste trabalho, e será considerada em trabalhos futuros.

## 6. Equilíbrio na Interação Cliente-Servidor

Uma vez estabelecida a anatomia dos dispositivos e uma base teórica adequada para abordar as interações entre agentes, podemos avaliar o equilíbrio e a otimalidade das decisões individuais. Primeiramente, avaliaremos as ações ótimas de cada agente, compararemos a qualidade das estratégias sob a perspectiva de estratégias *Minimax* [Albrecht et al. 2024]

e, por fim, enunciaremos a emergência de um equilíbrio advindo do balanço entre ganhos e riscos de cada agente. Este trabalho considera um cenário de dois dispositivos independentes.

### 6.1. Oferta de Recursos do Servidor

Por simplicidade, modelaremos o servidor como um dispositivo que atende uma requisição por vez, de modo a elevar ao máximo as consequências de um *offloading* bem-sucedido, transformando a situação de colisão de acesso em um jogo (em forma normal e de um único turno) de soma zero. Em caso de concorrência de acesso ao servidor, cada dispositivo de borda $i$ terá uma probabilidade de sucesso de conexão denotada por $\mathcal{P}_i = \mathbb{P}[\text{sucesso} \mid \text{emissor} = i]$. O servidor sempre aceita exatamente uma das requisições. O dispositivo que falhar na transmissão receberá uma confiança de nuvem igual a $0$, incorrendo apenas nos prejuízos energéticos do *offloading*, conforme Eqs. (1) e (6). Precisamos agora observar a perspectiva tática de cada agente ao realizar um *offloading*.

### 6.2. Retorno Esperado e Melhor Resposta

Conforme estabelecido pela definição de arrependimento (4), os agentes tomam decisões visando retornos esperados e ótimos ao longo do processo decisório. Dado um histórico de execução composto por estados e ações dos agentes nos turnos anteriores, denotado por $\mathbf{h}_t$, podemos definir o retorno esperado para o agente $i$:

$$U_i(\boldsymbol{\pi}_t) = \mathbb{E}[u_i(\mathbf{h}_t)], \tag{5}$$

em que $\boldsymbol{\pi}_t = (\pi_{1t}, \pi_{2t})$ é um *perfil de políticas*, ou de *estratégias*.

A cada turno, os dispositivos recebem oportunidades de melhoria para suas políticas de *offloading* a partir do mecanismo de *feedback*. Os *feedbacks* geram oportunidades de ação que serão avaliadas na iteração do algoritmo UCB do agente $i$, de acordo com a seguinte definição de *melhor resposta* (*Best Response*): $\mathrm{BR}_{it}(\boldsymbol{\pi}_{-it}) = \mathrm{argmax}_{\pi_{it}} U_i(\pi_{it}, \boldsymbol{\pi}_{-it})$, isto é, trata-se da política que maximiza o retorno esperado para o dispositivo $i$, levando em consideração as políticas dos demais dispositivos $\boldsymbol{\pi}_{-it}$. De acordo com o critério de seleção de limiares do UCB, a melhor resposta prescreve que probabilidade de *offloading* seja ajustada gradualmente. As buscas dos agentes por limiares ótimos dão origem a estratégias que, em emergência, levam a fenômenos de equilíbrio.

### 6.3. Estratégias Minimax e Equilíbrio de Nash

O equilíbrio em sistemas competitivos como o proposto implica um escalonamento automático dos dispositivos, de modo a balancear a oferta de recursos e a demanda por processamento. Este fato não ocorre por acaso, visto que o equilíbrio estudado é uma emergência das estratégias individuais em busca da otimalidade de seus limiares.

Uma estratégia automaticamente aprendida pelos agentes ao longo da execução do jogo estocástico é a denominada *Minimax*. Trata-se de escolher uma política que minimize os riscos de piores cenários. Isso é feito de forma automática para limiares trivialmente ruins com a informação adquirida ao longo de $T$ turnos, mas também se aplica a limiares sob ponderação do agente (que ainda possuem chances de serem avaliados em rodadas futuras). Formalmente, dado um perfil de políticas $\boldsymbol{\pi}_t = (\pi_{1t}, \pi_{2t})$, podemos descrever a estratégia *Minimax* para o dispositivo $i$ como:

$$U_i(\boldsymbol{\pi}_t) = \min_{\pi'_j} \max_{\pi'_i} U_i(\pi'_i, \pi'_j), \tag{6}$$

ainda que o dispositivo não tenha ciência da existência de $j$. Trata-se de uma estratégia que busca precaução contra um cenário adversarial sobre o qual o agente aprende interativamente. No caso do UCB, isso ocorre ao evitar fronteiras inferiores de confiança mais baixa, a menos que isso seja compensado por potenciais ganhos elevados, o que é mais comum no início da execução do algoritmo. Conforme o agente adquire maturidade, as informações adicionais contidas no histórico passam a ter maior peso na decisão, e os riscos são melhor gerenciados.

Ao atingir um grau elevado de experiência, ambos os agentes equilibram suas melhores respostas a partir das estratégias *Minimax*, resultando no chamado *Equilíbrio de Nash*. Um perfil de políticas $\boldsymbol{\pi}_t = (\pi_{1t}, \pi_{2t})$ é um equilíbrio de Nash quando nenhum dispositivo consegue melhorar seus retornos esperados alterando unilateralmente sua própria estratégia. Ou seja:

$$\forall i, \forall \pi'_{it} : U_i(\pi'_{it}, \boldsymbol{\pi}_{-it}) \leq U_i(\pi_{it}, \boldsymbol{\pi}_{-it}). \tag{7}$$

Isso implica que, no cenário analisado, cada dispositivo terá convergido para uma distribuição ótima de probabilidade de *offloading*, dadas as incertezas da rede e a capacidade energética disponível. Embora possa haver uma discrepância entre as estratégias ótimas (do ponto de vista sistêmico) e aquelas obtidas por meio de um Equilíbrio de Nash, a existência de tal equilíbrio possui implicações significativas, tais como a viabilização de um escalonamento descentralizado e gradual via aprendizado dos agentes.

### 6.4. Equilíbrio Aproximado

As limitações de representação finita de números reais e a eventual necessidade de longos episódios para a estabilização dos equilíbrios nos levam à necessidade de adotar graus de tolerância para as melhores respostas mútuas dos dispositivos. Para tal, pode-se relaxar a condição de equilíbrio (7) para incluir um fator de flexibilidade $\epsilon$:

$$\forall i, \forall \pi'_{it} : U_i(\pi'_{it}, \boldsymbol{\pi}_{-it}) \leq U_i(\pi_{it}, \boldsymbol{\pi}_{-it}) + \epsilon. \tag{8}$$

Graus de flexibilidade mais restritos ou generosos podem ser atribuídos em função da necessidade (como balancear riscos ou estabelecer margens de segurança na alocação de recursos) e obtidos ao avaliarmos a natureza dos algoritmos, as distribuições de confiança e a capacidade computacional do dispositivo. Na seção seguinte, discorremos sobre os resultados experimentais que revelam comportamentos de longo prazo e o impacto do perfil energético dos dispositivos na construção de estratégias e equilíbrios.

## 7. Resultados Experimentais

Os experimentos neste artigo consideram um cenário simples com dois dispositivos móveis, fixando as distribuições de confiança intermediária na borda e confiança da última camada na nuvem.[3]

[3]Um kit de reprodução editável, com aplicação web para simulações customizadas e de distribuição livre para os experimentos pode ser consultado em `https://github.com/selsoaress/SBRC-2026-Offloading-Adaptativo`.

As funções e os parâmetros adotados nos experimentos foram definidos de modo a capturar os principais aspectos do modelo proposto. O custo energético associado ao *offloading* é modelado por uma função de feedback negativo proporcional ao número de transmissões realizadas pelo dispositivo, variando de forma linear em relação à energia investida. Em particular, essa função é dada por $\eta_j(o_{jt}) = (\eta_{j,\max} - o_{j,t} \cdot \mu_c)/\eta_{j,\max}$, onde $\mu_c$ representa o consumo energético unitário esperado. As confianças obtidas na borda e na nuvem são modeladas por distribuições Beta, sendo a confiança na borda descrita por uma distribuição $\mathrm{Beta}(8, 4)$, enquanto a confiança após o processamento em nuvem segue uma distribuição $\mathrm{Beta}(12, 3)$. Os parâmetros específicos de cada dispositivo de borda são detalhados nas legendas dos gráficos correspondentes aos cenários analisados. Todas as simulações foram executadas em 40 repetições independentes, de modo a obter métricas de desempenho médio para uma análise mais robusta. Para suavização gráfica, são realizadas médias móveis (100 últimos turnos) a cada ponto. Avaliamos dois cenários:

- Cenário A: o dispositivo $D_1$ possui simultaneamente maior preferência de serviço, ou seja, maior probabilidade de obter os recursos da nuvem, e portanto, realizar *offloading* e maior capacidade energética em termos de bateria;
- Cenário B: o dispositivo $D_1$ ainda possui maior preferência de serviço da nuvem, mas possui menor capacidade energética que o dispositivo $D_2$.

### 7.1. Evolução de Limiares e Equilíbrios

A Figura 2 ilustra a evolução dos limiares de confiança selecionados pelos dispositivos $D_1$ e $D_2$ ao longo das interações do jogo, considerando diferentes combinações de preferência por serviço — isto é, a probabilidade de acesso aos recursos da nuvem — e capacidade energética. Em particular, a Figura 2(a) apresenta os resultados do Cenário A, no qual o dispositivo $D_1$ detém simultaneamente maior preferência de serviço e vantagem energética. Por sua vez, a Figura 2(b) corresponde ao Cenário B, em que o dispositivo $D_1$ possui menor capacidade energética, embora mantenha maior preferência pelo serviço. Em ambos os cenários, observa-se uma variação mais acentuada dos limiares durante a fase inicial de exploração pura, correspondente aos 100 primeiros turnos. Contudo, o comportamento exploratório não é interrompido abruptamente, sendo evidenciado pela persistência de oscilações dos limiares dentro de uma faixa progressivamente mais estreita ao longo das iterações subsequentes.

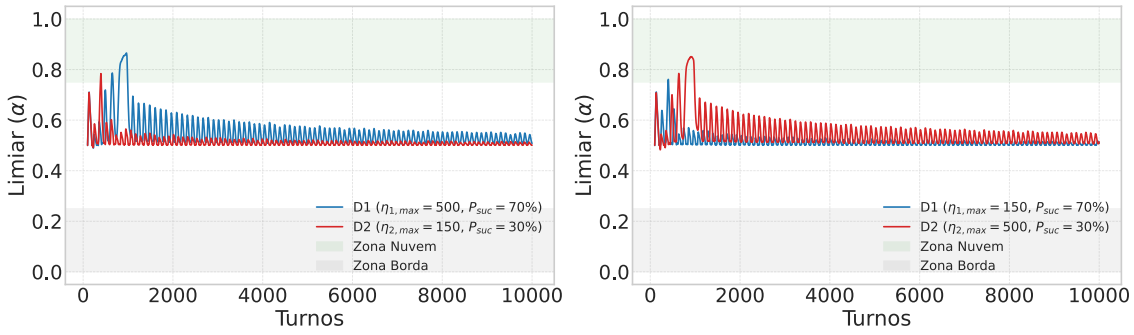


**Figura 2. Dinâmica de ajuste dos limiares de confiança ($\alpha$) ao longo dos turnos, nos dois cenários avaliados.**

**Impacto da vantagem energética na exploração de limiares:** Uma maior capacidade energética por parte de um dispositivo confere uma exploração mais rica de limiares ao sistema agraciado, fornecendo orçamento suficiente para avaliar estratégias mais agressivas ou comedidas, a despeito das prioridades de acesso do dispositivo com o qual compartilha recursos. No início da simulação, nota-se uma exploração mais enérgica para a descoberta de padrões, consolidando a escolha de limiares mais intensos para os dispositivos com maior reserva de energia. No entanto, a longo prazo, a tendência é de estabilização, com predominância do dispositivo mais potente.

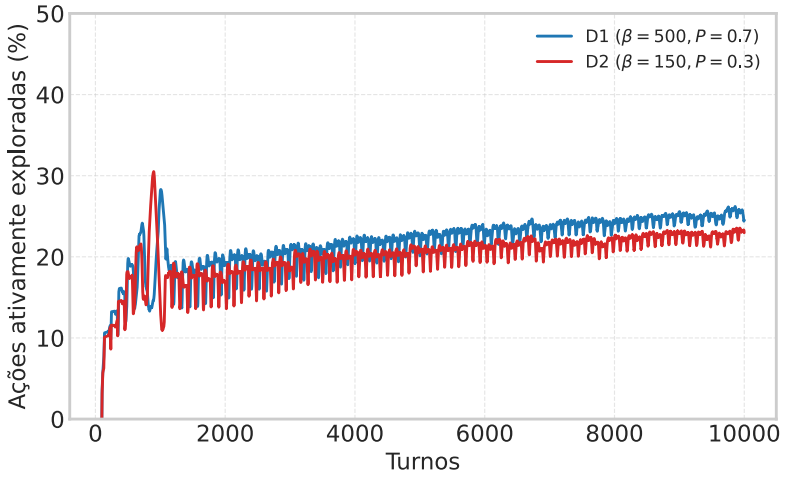


**Figura 3. Desvio padrão dos limiares com base em 100 rodadas anteriores. Quanto mais próximo de 0, maior é a proximidade de convergências de Nash.**

**Relação entre variação de limiares explorados e equilíbrios aproximados:** Os *trade-offs* do paradigma de *multi-armed bandits* entre exploração e explotação (*exploration vs. exploitation*) podem ser observados (Figura 3) conforme a análise da proporção de ações ativas entre os dispositivos. Em geral, observa-se que cerca de 20% das opções de limiares são mantidas no cenário analisado. Tal fenômeno vincula-se ao conceito de equilíbrios aproximados, visto que existe uma margem de incerteza implícita no conjunto reduzido de opções; assim, o agente mantém uma lista de candidatos promissores em vez de convergir precocemente para um ótimo global estático.

### 7.2. Evolução das Recompensas Médias e Consumo Energético

As recompensas são diretamente impactadas pelas escolhas de limiares. A ação de classificação na borda assume um papel central para dispositivos que enfrentam dificuldades de conexão com o servidor. O agente maximiza as recompensas ao ponderar continuamente o risco, variando os limiares dentro da faixa de equilíbrio sob avaliação.

**Prioridades de Acesso e Assiduidade:** Um impacto direto da vantagem do agente se manifesta na sua capacidade de manter uma assiduidade de uso dos servidores, o que contribui para que um retorno médio acumulado mais elevado seja obtido para um mesmo perfil energético (ver Figura 4).

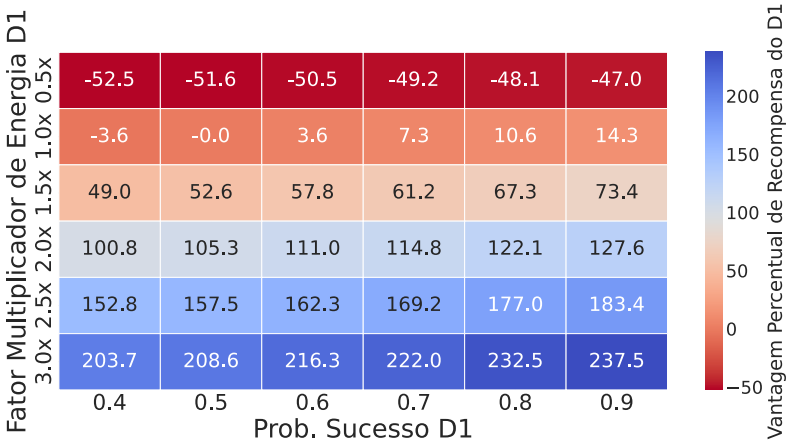


**Figura 4. Vantagem (em %) de recompensas médias acumuladas pelo dispositivo D1 ao longo das simulações para diferentes perfis energéticos e de prioridade de acesso.**

## 8. Conclusões e Próximos Passos

Este trabalho propôs um ajuste descentralizado e online do limiar de confiança $\alpha$ em EENNs via MAB/UCB, modelando a competição entre dispositivos por recursos de nuvem como um jogo estocástico com informação parcial e restrições energéticas. Os experimentos indicam que heterogeneidade de prioridade de acesso e de bateria altera de forma sistemática a dinâmica de $\alpha$, e que a redução da variabilidade do limiar ao longo do tempo é consistente com a estabilização em um equilíbrio aproximado. Como próximos passos, pretendemos estender o modelo para múltiplos *early exits* e cenários não-estacionários, além de validar a abordagem com modelos e dispositivos reais incorporando latência/filas na infraestrutura de nuvem.

## Referências

Albrecht, S. V., Christianos, F., and Schäfer, L. (2024). *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*.

Altman, E. (1999). *Constrained Markov Decision Processes*, volume 7 of *Stochastic Modeling*. Boca Raton, FL.

Bajpai, D. J. and Hanawal, M. K. (2025). BEEM: Boosting performance of early exit DNNs using multi-exit classifiers as experts. In *International Conference on Learning Representations (ICLR)*.

Casale, G. and Roveri, M. (2023). Scheduling inputs in early exit neural networks. *IEEE Transactions on Computers*, 73(2):451–465.

Fang, B., Zeng, X., Zhang, F., Xu, H., and Zhang, M. (2020). Flexdnn: Input-adaptive on-device deep learning for efficient mobile vision. In *IEEE/ACM Symposium on Edge Computing (SEC)*, pages 84–95.

Ju, W., Bao, W., Ge, L., and Yuan, D. (2021a). Dynamic early exit scheduling for deep neural network inference through contextual bandits. In *International Conference on Information Knowledge Management (CIKM)*, pages 823–832.

Ju, W., Bao, W., Yuan, D., Ge, L., and Zhou, B. B. (2021b). Learning early exit for deep neural network inference on mobile devices through multi-armed bandits. In *IEEE International Symposium on Cluster, Cloud, and Internet Computing (IEEE/ACM CCGrid)*, pages 11–20.

Kim, G. and Park, J. (2020). Low cost early exit decision unit design for cnn accelerator. In *IEEE International SoC Design Conference (ISOCC)*, pages 127–128.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Laskaridis, S., Venieris, S. I., Almeida, M., Leontiadis, I., and Lane, N. D. (2020). SPINN: synergistic progressive inference of neural networks over device and cloud. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 1–15.

Li, E., Zeng, L., Zhou, Z., and Chen, X. (2019). Edge ai: On-demand accelerating deep neural network inference via edge computing. *IEEE Transactions on Wireless Communications*, 19(1):447–457.

Liu, C., Liu, K., Guo, S., Xie, R., Lee, V. C., and Son, S. H. (2020a). Adaptive offloading for time-critical tasks in heterogeneous internet of vehicles. *IEEE Internet of Things Journal*, 7(9):7999–8011.

Liu, Y., Peng, M., Roedig, U., and Rodrigues, J. J. (2020b). Vehicular edge computing and networks: A survey. *Mobile Networks and Applications*, 25:1157–1168.

Mao, Y., You, C., Zhang, J., Huang, K., and Letaief, K. B. (2017). A survey on mobile edge computing: The communication, computation, and energy perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358.

Pacheco, R. G., Bajpai, D. J., Shifrin, M., Couto, R. S., Menasché, D. S., Hanawal, M. K., and Campista, M. E. M. (2024). Ucbee: A multi armed bandit approach for early-exit in neural networks. *IEEE Transactions on Network and Service Management*.

Pacheco, R. G., Bajpai, D. J., Shifrin, M., Couto, R. S., Menasché, D. S., Hanawal, M. K., and Campista, M. E. M. (2025). Otimizando saídas antecipadas em redes neurais profundas: Como lidar com buffers? In *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, pages 560–573.

Pacheco, R. G., Couto, R. S., and Simeone, O. (2021). Calibration-aided edge inference offloading via adaptive model partitioning of deep neural networks. In *IEEE International Conference on Communications (ICC)*, pages 1–6.

Rahmath P., H., Srivastava, V., Chaurasia, K., Pacheco, R. G., and Couto, R. S. (2024). Early-exit deep neural network-a comprehensive survey. *ACM Computing Surveys*, 57(3):1–37.

Satyanarayanan, M. (2017). The emergence of edge computing. *Computer*, 50(1):30–39.

Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646.

Slivkins, A. et al. (2019). Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning*, 12(1-2):1–286.

Teerapittayanon, S., McDanel, B., and Kung, H.-T. (2016). Branchynet: Fast inference via early exiting from deep neural networks. In *IEEE International Conference on Pattern Recognition (ICPR)*, pages 2464–2469.

Wang, M., Mo, J., Lin, J., Wang, Z., and Du, L. (2019a). Dynexit: A dynamic early-exit strategy for deep residual networks. In *IEEE International Workshop on Signal Processing Systems (SiPS)*, pages 178–183.

Wang, Z., Bao, W., et al. (2019b). SEE: Scheduling early exit for mobile dnn inference during service outage. In *ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, pages 279–288.

Zamzam, M., El-Shabrawy, T., and Ashour, M. (2020). Game theory for computation offloading and resource allocation in edge computing: A survey. In *IEEE Novel Intelligent and Leading Emerging Sciences Conference (NILES)*, pages 47–53.