https://doi.org/10.7840/kics.2024.49.6.839

통신 에러를 고려한 스플릿 레이어 선택 기법

이 재 욱•, 고 근 수*, 고 한 얼°

Transmission Error Aware Split Layer Selection Scheme

Jaewook Lee•, Keunsoo Ko*, Haneul Ko°

요 약

스플릿 컴퓨팅을 위한 많은 연구들은 단말이 하위 모델을 처리한 후 에지 서버에게 자신의 하위 모델 처리 결과를 전달할 때 발생하는 통신 에러를 고려하지 않고 있는 문제점이 존재한다. 즉, 전송 에러로 인해 에지 서버는 본래의 중간 결과 값과는 다른 데이터를 기반으로 나머지 하위 모델을 처리하기 때문에 추론 결과의 성능이 떨어지는 문제가 발생한다. 따라서, 본 논문에서는 통신 에러가 존재하는 환경에서 스플릿 컴퓨팅 기법의 성능을 실험적을 분석하고, 서비스의 품질을 최대화 시키는 스플릿 레이어 선택 기법인 transmission error aware split layer selection (TESLS) 기법을 제안한다. TESLS 기법은 사전지식없이 환경과 상호작용하여 스플릿 레이어 기법에 따른 성능 관계를 배우고 최적의 스플릿 레이어를 선택할 수 있도록 고안되었다. 본 기법은 다양한 실험을 통해 우수성을 입증하였다.

키워드 : 스플릿 컴퓨팅, 통신 에러, 다중 팔 문제, AI 서비스
Key Words : Split computing, transmission error, multi-armed bandit problem, AI service

ABSTRACT

Most of the works for improving split computing service performance did not consider the transmission error when the IoT device transmits the processing result of its sub-model (i.e., intermediated data) to the edge server. Unfortunately, the transmission error can degrade the inference result of the whole deep model due to the erroneous intermediated data. Thus, in this paper, we first analyze the effect of transmission error on the split computing service performance based on the simulation study. Then, we propose transmission error aware split layer selection (TESLS) to improve the service quality. In the TESLS scheme, the service controller interacts with the environment to find the best-split layer. The simulation results confirm that our proposed work is outperformed to the conventional split layer selection scheme.

Ⅰ. 서 론

최근 인공지능/기계학습 (artificial intelligence/machine learning, AI/ML) 기법의 비약적인 발전으로 인해, 많은 서비스 어플리케이션들이 인공지능 기반 (즉, 심층 모델 기반)으로 구현되고 있으며, 이러한 심층 모델들은 더 좋은 서비스 품질을 제공하기 위해 많은 컴퓨팅 연산 작업을 요구하고 있다[1]. 이러한 특징은 Internet of Things (IoT) 단말들과 같이 제약적인 컴퓨팅 성능 갖는 디바이스들이 실시간으로 심층 모델을 처

※ 본 연구는 부경대학교 자율창의학술연구비(2023~2024년)에 의하여 연구되었습니다.
• First Author : Pukyong National University Department of Information and Communication Engineering, jlee0315@pknu.ac.kr, 정회원
° Corresponding Author : Kyung Hee University Department of Electronic Engineering, heko@khu.ac.kr, 종신회원
* Catholic University of Korea, Department of Artificial Intelligence, ksko@catholic.ac.kr,
논문번호 : 202312-160-C-RN, Received November 30, 2023; Revised January 17, 2024; Accepted February 11, 2024

리하기 어렵게 만든다[1-3].

제약적인 컴퓨팅 성능을 갖는 디바이스들이 실시간으로 심층모델을 처리하고자 스플릿 컴퓨팅 (Split Computing) 서비스가 등장하였고, 해당 서비스는 미래 시대의 주요한 네트워크 서비스로 고려되고 있다[1-5]. 스플릿 컴퓨팅에서는 처리할 심층 모델을 스플릿 레이어를 기준으로 두개의 작은 모델로 나눈다. 나뉘어진 두개의 모델 중 앞 부분의 심층 모델 (i.e., Head Model)은 단말에서 처리하고, 뒷부분의 심층 모델 (i.e., Tail Model)은 에지 서버에서 처리된다. 즉, 단말들은 자신이 수집한 입력 값을 통해 Head Model을 먼저 추론하고, 추론된 결과값 (intermediate data)을 에지 서버에게 전송한다. 에지 서버는 단말들로부터 받은 추론 결과값을 기반으로 Tail Model을 추론하여, 인공지능 서비스의 결과값을 획득한다. 끝으로, 에지 서버가 결과값을 단말에게 다시 전송함으로써 인공지능 기반의 어플리케이션 작업이 완료된다. 이처럼, 단말이 전체 심층 모델이 아닌 일부 심층 모델만 처리함으로써 단말의 컴퓨팅 부담을 효과적으로 줄일 수 있으며, 단말이 수집한 원본 입력 데이터가 아닌 가공된 데이터가 에지 서버에 전송하므로, 단말의 개인 정보를 보호할 수 있다. 또한, 컴퓨팅 자원이 비교적 풍부한 에지 서버가 심층 모델 일부를 대신 추론해주기 때문에 실시간으로 서비스를 처리할 수 있다.

스플릿 컴퓨팅 서비스 성능은 선택된 스플릿 레이어에 따라 서비스 완료 시간이 결정되는 특징이 존재한다. 이에, 많은 연구들 에서는 더 빠른 서비스 완료 시간을 달성하기 위한 스플릿 레이어 선택 기법에 대한 연구를 주로 수행했다[6-8]. 하지만, 해당 연구들은 무선 통신상에 오류가 없는 이상적인 통신상태를 가정하였고, 몇몇의 연구에서만[9] 통신 오류에 강건한 스플릿 컴퓨팅 서비스에 대해 연구하였다. [9]에서 제시한 실험 결과를 보면, 전송 오류에 의해 중간 결과값에 오류가 발생하면 서비스 추론 정확도가 저하되는 것을 확인할 수 있다. 이러한 문제를 해결하고자, [9]에서는 전송 오류율에 따른 재전송 횟수를 조정하는 기법을 제안하였다. 하지만 제약적인 에너지를 갖는 IoT 단말들에게는 재전송 기법은 에너지 소비가 많은 기술이며, 재전송 횟수를 정하기 위해서는 셀룰러 기지국과 단말의 셀룰러 스택을 수정해야 하는 현실적인 문제가 존재한다.

본 논문에서는 전송 오류에 따른 IoT 서비스 품질 (서비스 완료 시간 및 서비스 추론 정확도)의 변화를 실험적으로 분석하고, 분석된 결과를 토대로 전송상에 오류가 발생하는 환경에서도 서비스 품질을 최대화하는 실용적인 스플릿 레이어 선택기법인 transmission error aware split layer selection (TESLS) 기법을 제안한다. TESLS 기법은 multi-armed bandit 기술[10,11]을 통해 사전 지식이 없이 실시간으로 스플릿 레이어에 따른 서비스 품질의 변화를 경험적으로 배우고, 최적의 스플릿 레이어를 선택할 수 있도록 설계하였다. 따라서 본 기법은 기존의 네트워크 인프라 (셀룰러 스택 등) 수정없이 어떠한 환경에서도 실용적으로 적용 가능하다. TESLS 기법의 우수성을 입증하고자 다양한 실험을 수행하였으며, 실험을 통해 제안한 기법이 사용자가 요구하는 서비스 품질을 만족하는 스플릿 레이어를 선택함으로써 기존의 서비스 완료 시간만 최소화하는 기법보다 서비스 품질을 향상시키는 것을 확인하였다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 가정하는 시스템 모델을 소개한다. 그 후, 3장에서는 통신 에러를 고려한 스플릿 레이어 선택 기법을 제안한다. 본 장에서는 우선 통신 에러에 따른 스플릿 컴퓨팅 서비스의 품질에 대한 영향력을 분석하고, 스플릿 레이어 선택 기법인 TESLS 기법을 제안한다. 4장에서는 실험 환경과 결과를 설명하고 5장에서 본 논문을 끝맺는다.

Ⅱ. 시스템 모델

그림 1은 본 논문에서 가정하는 에지 클라우드가 배포된 무선 IoT 네트워크를 나타낸다. 해당 네트워크에는 IoT 단말들이 배포되어 있으며, IoT 단말들은 기지국 (access point, AP)을 통해 에지 서버와 통신할 수 있다. IoT 단말들은 심층 모델 $\boldsymbol{D}=(\boldsymbol{L}, \boldsymbol{O}, \boldsymbol{C})$ 를 스플릿 컴퓨팅 서비스를 통해 추론하고자 한다. 해당 서비스는 학습이 완료된 심층 모델을 추론함으로써 사용자에게 제공된다. 심층 모델 $\boldsymbol{D}$에서 $\boldsymbol{L}$, $\boldsymbol{O}$ 그리고 $\boldsymbol{C}$는 심층 모델의 심층 레이어 인덱스에 대한 집합, 각 심층 레이어의 출력 데이터 크기들의 집합 그리고 각 심층 레이어를

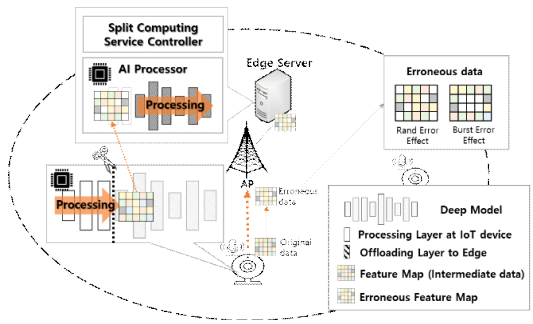


그림 1. 시스템 모델
Fig. 1. System Model.

연산하기 위한 요구하는 연습량들을 나타내는 집합이다. 심층 레이어 인덱스 집합 $\boldsymbol{L}$ 은 {0, 1, ⋯, L}로 정의되면 0 번째와 L번째 레이어들은 입력 레이어와 출력 레이어를 각각 나타낸다. 또한, 심층 레이어의 출력 데이터 크기 집합 $\boldsymbol{O}$는 {o_0, o_1, ⋯, o_l, ⋯, o_L}로 나타내며, 집합의 원소 o_l는 l번째 심층 레이어의 출력 데이터 크기를 나타낸다. 심층 레이어 연산 량 집합 $\boldsymbol{C}$는 {c_0, c_1, ⋯, c_l, ⋯, c_L}로 표현할 수 있으며, c_l은 l번째 심층 레이어를 연산하기 위한 연산 량을 나타낸다.

에지 서버에는 스플릿 컴퓨팅 서비스를 제어하기 위한 스플릿 컴퓨팅 서비스 컨트롤러가 소프트웨어로 구현되어 있으며, 해당 컨트롤러에서는 IoT 단말의 컴퓨팅/통신 성능 등을 수집하고 타겟 IoT 서비스를 수행하기 위한 심층 모델의 스플릿 레이어를 선택하고, 선택된 스플릿 레이어를 IoT 단말에게 알려주는 역할을 수행한다. 이때, 서비스의 품질은 전체 심층 모델이 처리되는 서비스 완료 시간과 서비스 추론 정확도 두 값을 통해 표현된다. 만일 컨트롤러가 t시점에서 l번째 심층 모델을 스플릿 레이어로 결정하면, 스플릿 컴퓨팅 서비스를 통해 달성되는 서비스 처리 시간 $d_{l,t}$은 수식 (1) 과 같이 정의된다.

$$d_{l,t} = \frac{\sum_{l'=0}^{l'=l} c_{l'}}{f_{\mathrm{I},t}} + \frac{o_l}{b_{\mathrm{I},t}} + \frac{\sum_{l'=l}^{l'=L} c_{l'}}{f_{\mathrm{E},t}} \quad (1)$$

수식 (1)에서 $f_{\mathrm{I},t}$ 와 $f_{\mathrm{E},t}$는 t 시점에서 IoT 단말의 컴퓨팅 연산 능력과 에지 서버의 컴퓨팅 연산 능력을 나타낸다. 또한, $b_{\mathrm{I},t}$는 t 시점에서 IoT 단말의 통신 처리 능력을 나타낸다. 반면에, 같은 상황에서 서비스 추론 정확도 $acc_{\mathrm{I},t}$는 수식 (2) 과 같이 나타낸다.

$$acc_{l,t} = h\left(\mathrm{E}_{\mathrm{type},t}, \mathrm{E}_{\mathrm{rate},t}, l\right) \quad (2)$$

수식 (2)에서 $\mathrm{E}_{\mathrm{typt},t}$와 $\mathrm{E}_{\mathrm{rate},t}$는 t 시점의 무선 네트워크에 에러 타입과 에러율을 나타낸다. 에러 타임은 [9]와 같이 랜덤 에러 타입 (Random) 과 버스트 에러 타입 (Burst)이 존재하고, 에러율은 전송에 의한 bit error rate (BER)을 고려한다. h 함수는 입력에 의한 서비스 정확도를 나타내는 함수이다.

스플릿 컴퓨팅의 서비스 품질은 심층 모델의 추론 완료 시간과 추론 정확도를 모두 고려하여 도출된다. 심층모델의 추론 완료 시간은 수식 (1)을 통해 스플릿 레이어를 선택하기 전에 도출할 수 있지만, 수식 (2)의 h 함수를 수학적으로 정의하기 어렵기 때문에 심층모델의 추론 정확도를 스플릿 레이어를 선택하기 전에 컨트롤러가 알기 어려운 문제점이 존재한다. 따라서, Ⅲ 장에서는 컨트롤러가 사전지식이 없이 스플릿 레이어에 따른 추론 완료 시간과 추론 정확도를 배우고, 최적의 스플릿 레이어를 선택하는 기법인 TESLS 기법을 제안한다. 해당 기법을 제안하기전에 우선 통신 에러에 따른 서비스 품질 (서비스 추론 정확도 및 서비스 완료 시간)에 대한 관계를 실험적으로 분석한다.

Ⅲ. 통신 에러를 고려한 스플릿 레이어 선택 기법

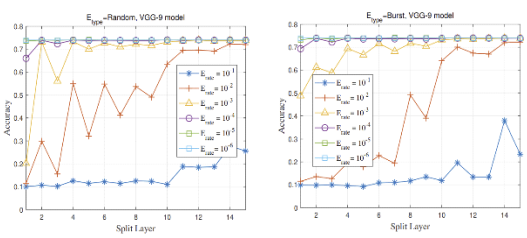
본 장에서는 통신 에러에 따른 스플릿 컴퓨팅 서비스 성능 (추론 정확도 및 서비스 처리 완료 시간)의 변화 먼저 실험적으로 분석하고, 통신 에러를 고려한 스플릿 레이어 선택 기법을 제안한다.

3.1 통신 에러에 따른 스플릿 컴퓨팅 서비스 성능 분석

수식 (2)의 h함수의 경향성을 실험적으로 확인하기 위해, 즉 통신 에러에 따른 스플릿 컴퓨팅 서비스의 추론 정확도를 실험적으로 분석하기 위해 단말이 송신하는 중간 데이터 중 일부의 데이터가 통신 에러에 의해 소실된 데이터 (즉, 에러가 반영된 중간데이터)를 에지 서버에게 전달되는 테스트 환경을 구성하였다. 본 실험에서 통신 에러율은 [10^{-1}~10^{-7}]의 범위 내의 bit error rate로 고려하였다[9]. 본 실험을 위해 사용된 심층 모델은 분류 문제에 가장 널리 사용되는 VGG 형태의 심층 모델을 사용하였으며, Cifar10 데이터를 통해 학습하고 평가하였다.

3.1.1 스플릿 레이어 및 통신 에러율에 따른 성능 분석

그림 2(a) 와 (b)는 각 스플릿 레이어에서 통신 에러가 random 에러 타입인 경우와 burst 에러 타입인 경우에 추론 정확도를 각각 나타낸다. 그림 2(a) 와 (b)의 결과에서는 공통적으로 앞부분의 레이어가 스플릿 레이어로 선택되는 경우 에러율에 따라 추론 정확도의 감



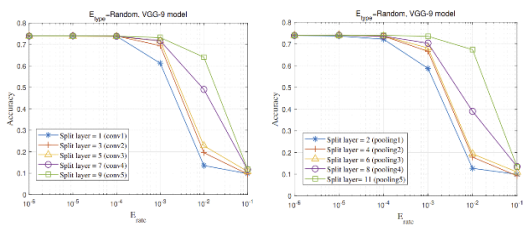
(a) (b)

그림 2. 통신 에러율 및 에러 종류에 따른 영향
Fig. 2. Effect of error rate and error type

소가 크다는 것을 확인할 수 있다. 이는 앞부분의 레이어로부터 축출된 입력 값은 분류를 수행하는데 있어 중요한 특징들이고, 이러한 특징들이 오염되었기 때문에 분류를 정확히 수행하지 못하는 문제가 발생하기 때문이다. 반면에 그림 2(a)와 (b)를 비교해보면, 전반적으로 burst 에러 타입이 발생하는 경우 random 에러 타입이 발생하는 경우보다 추론 정확도를 더 감소시키는 것을 확인하였다. Burst 에러 타입의 경우 비슷한 공간의 정보들이 누락되기 때문에 이를 복원하는데 어려움이 존재한다. 이와 반면에, random한 에러타입의 경우 공간적으로 분산된 정보들이 누락되기 때문에 오염되지 않은 근처의 정보들로 비교적 쉽게 복원이 가능하다. 이러한 특징으로 인해 burst한 에러 타입이 발생하는 경우 서비스 추론 성능에 더 좋지 못한 영향을 발생시킨다. 그림 2 (a)와 (b)의 결과값이 톱니 형태로 나타나는 이유는 컨벌루젼 레이어와 풀링 레이어가 번갈아 가면서 존재하기 때문이다.

3.1.2 심층 모델 크기에 따른 성능 분석

그림 3(a)와 (b)는 스플릿 레이어가 컨블루젼 레이어일 때와 풀링 레이어 일 때의 심층 모델의 추론 정확도 성능을 나타낸다. 그림 3(a)와 (b)의 결과 값에서 공통적으로 레이어의 종류와 상관없이 후반부의 레이어를 스플릿 레이어로 선택한 경우 더 좋은 추론 정확도를 보여주는 것을 확인 할 수 있다. 반면에, 그림 3(a)와 (b)를 비교해보면 스플릿 레이어가 컨볼루젼 레이어인 경우 풀링 레이어 일때보다 더 좋은 추론 정확도를 나타낸 것을 확인할 수 있다. 이는 풀링 레이어에서 중요한 특징들을 압축하여 적은 데이터로 표현한 결과값을 제공하기 때문에 해당 정보들이 조금이라도 오류가 발생하면 압축된 정보들이 모두 오류가 발생하는 효과가 발생한다. 이러한 특징으로 풀링 레이어의 결과값에 오류가 발생하는 경우 추론 정확도에 더 안 좋은 영향을 끼친다. 이와 반면에 컨블루젼 레이어에서는 정보를 가공할 뿐 정보의 양을 줄이지 않기 때문에 비교적 전송



(a) (b)

그림 3. 스플릿 레이어에 따른 영향
Fig. 3. Effect of split layer

오류에도 강건한 특징을 갖게 된다.

3.1.3 스플릿 레이어에 따른 추론 시간

그림 4는 스플릿 레이어 따른 서비스 추론 시간과 각 요소별로의 추론 시간들을 나타낸다. 추론 시간은 재전송이 없는 한 전송오류에 영향을 받지 않는다. 그림 4를 통해 스플릿 레이어가 후반부 레이어로 결정될수록 단말이 처리하는 연산량이 많게 되는 (즉, 단말이 오랫동안 연산을 수행하는)것을 확인할 수 있고, 이러한 특성으로 서비스 처리 완료시간이 증가하는 것을 확인할 수 있다. 그림 4의 그래프 또한 톱니 바퀴형태의 형상을 보여주는데 이는 풀링 레이어를 스플릿 레이어로 선택하는 경우 중간 데이터양을 줄여 전송 시간을 획기적으로 감소시킬 수 있기 때문이다.

그림 2, 3 그리고 4의 결과를 비교해보면 서비스 처리완료시간은 스플릿 레이어가 앞부분 레이어 면서 풀링 레이어 일 때 가장 큰 이득을 볼 수 있지만, 서비스 추론 정확도는 뒷부분의 컨볼루젼 레이어가 스플릿 레이어일 때 통신 에러에 강건한 특징을 보여준다. 또한, 통신 에러에 따른 추론 정확도의 경우 큰 경향성은 존재하나 수식적으로 정의 할 수 없는 (즉, 수식 (2)의 h함수를 정의 할 수 없는) 제약점을 실험적으로 확인하였다. 따라서, 최적의 스플릿 레이어를 선택하기 위해서는 스플릿 레이어에 따른 심층 모델의 추론 정확도의 관계를 배우고, 최종적으로 요구하는 서비스 처리 완료시간을 만족하면서 최대의 서비스 추론 정확도를 제공 할 수 있는 최적의 스플릿 레이어를 선택하는 기법에 관한 연구가 필요하다.

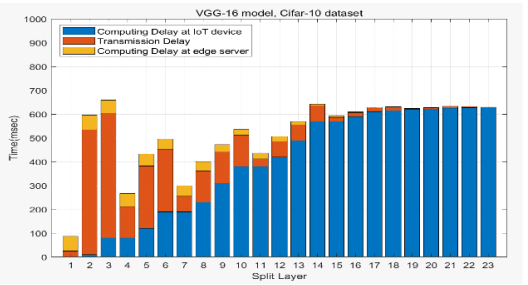


그림 4. 스플릿 레이어에 따른 추론 시간
Fig. 4. Effect of split layer on inference time

3.2 통신 에러를 고려한 스플릿 레이어 선택 기법

본 절에서는 요구하는 서비스 처리 완료 시간을 만족하면서 서비스 추론 정확도를 최대화하는 스플릿 레이어 선택 기법인 TESLS 기법을 제안한다. 3.1 절에서

분석한 바와 같이 통신 에러율과 타입 그리고 스플릿 레이어에 따른 정확도 (즉, 수식 (2)의 h함수)를 수학적으로 정의할 수 없다는 문제점 [12]을 해결하고자 TESLS기법은 경험을 통해 요구사항을 충족하는 스플릿 레이어를 찾아갈 수 있도록 Multi-Armed Bandit 기반으로 설계 되었다.

Multi-Armed Bandit 기법은 플레이어가 환경과 상호작용하여 가장 좋은 행위를 학습할 수 있다. 즉, 플레이어는 가능한 행위들 중에 하나를 수행하고 그에 따른 보상을 환경으로부터 습득한다. 해당 절차를 반복적으로 수행함으로써 플레이어는 각 행위에 대한 품질 (보상)를 학습하고, 최종적으로 보상이 가장 높은 최적의 행위를 찾을 수 있다. Multi-Armed Bandit 기법을 기반으로 TESLS 기법을 설계하기 위해서 선택 가능한 스플릿 레이어를 행위로 정의하였다. 따라서, TESLS 기법에서의 행위 집합 A는 다음과 같이 정의된다.

$$A = \{0,1,2,3,\ldots,L\} \tag{3}$$

또한, t 시점에 a 번째 심층 레이어가 스플릿 레이어로 결정된 경우 그에 따른 서비스 품질 $r_{a,t}$는 다음과 같이 정의된다[13].

$$r_{a,t} = acc_{a,t} \times \left[\frac{d_{a,t}}{d_{max}}\right]^{\omega} \tag{4}$$

수식 (4)에서 $d_{a,t}$는 t시점에 심층 레이어 a를 스플릿 레이어로 선택했을 때 서비스 처리 시간이 나타내고 d_{max}는 요구 서비스 처리 시간을 나타낸다. $acc_{a,t}$는 같은 상황에서의 서비스 추론 정확도를 나타낸다. $d_{a,t}$와 $acc_{a,t}$는 수식적으로 도출되는 값이 아니며, 경험적으로 단말들에게 획득할 수 있는 값이다. ω는 서비스 품질에서의 서비스 처리 시간에 대한 가중치이며 수식 (5)와 같이 정의된다[13].

$$\omega = \begin{cases} \alpha, & \text{if } d_{a,t} \le d_{max} \\ \beta, & \text{otherwise} \end{cases} \tag{5}$$

수식 (5)에서의 α와 β는 서비스 파라미터로 서비스마다 상의한 값이다.

정의된 행위와 보상 값을 통해 서비스 품질 값을 최대화하기 위한 최적화 문제는 수식 (6)과 같이 정의한다.

$$\max_{\pi(t)} \sum_{t=1}^{T} r_{\pi(t),t} \tag{6}$$

수식 (6)에서 $\pi(t)$는 t시점에 선택한 레이어를 나타내는 정책 함수이다. 알고리즘 1은 본 최적화 함수를 풀기 위해서 TESLS 알고리즘의 동작과정을 나타낸다. 우선, t번째 라운드까지 스플릿 서비스가 제공되는 동안 각 레이어가 스플릿 레이어로 선택된 횟수와 경험적으로 추측한 각 레이어의 서비스 품질 값을 나타내는 두개의 벡터 $k(t) = [k_0(t), \cdots, k_l(t), \cdots, k_L(t)]$와 $v(t) = [v_0(t), \cdots, v_l(t), \cdots, v_L(t)]$를 초기화 한다 (알고리즘 1에서 1번째 줄 동작과정). 여기서 $k_l(t)$와 $v_l(t)$는 t번째 라운드까지 심층 레이어 l이 스플릿 레이어로 선택된 횟수와 경험적으로 추측한 레이어 l의 서비스 품질 값을 나타낸다. T 라운드 동안 스플릿 서비스가 제공되며 알고리즘 1의 3번째 줄부터 11번째 줄까지의 명시된 동작과정들이 반복된다. t 라운드가 시작되면, 스플릿 컴퓨팅 서비스 컨트롤러는 심층 모델 내의 하나의 심층 레이어를 스플릿 레이어로 선택한다. 이때, 스플릿 컴퓨팅 서비스는 한 번도 선택되지 않은 레이어부터 우선적으로 선택하며 (알고리즘1에서 3~4번째 줄 동작과정), 모든 레이어가 한번 이상 선택된 경우 수식 (7)과 같은 기준으로 심층 레이어 하나를 선택한다 (알고리즘1에서 5~6번째 줄 동작과정).

$$a = \underset{a'}{\operatorname{argmax}}\left(v_a(t-1) + \sqrt{\frac{\alpha \ln t}{k_a(t-1)}}\right) \tag{7}$$

스플릿 컴퓨팅 서비스 컨트롤러는 결정된 스플릿 레이어 a를 에지 서버와 단말에게 알려 준 후, 에지 서버와 단말은 해당 레이어를 기준으로 스플릿 컴퓨팅 서비스를 수행한다. 단말은 해당 라운드의 스플릿 컴퓨팅

알고리즘 1. TESLS 알고리즘
Algorithm 1. TESLS Algorithm

Algorithm 1 TESLS algorithm.

1: Initialize vectors and sets $\mathbf{v}(t) = \mathbf{o}$ $\mathbf{k}(t) = \mathbf{o}$
2: **for all** t **do**
3: **if** $\min_a (k_a(t)) = 0$ **then**
4: $a = \underset{a'}{\operatorname{argmin}} (k_{a'}(t))$
5: **else if** $\min_a (k_a(t)) > 0$ **then**
6: $a = \underset{a'}{\operatorname{argmax}} (v_{a'}(t-1) - \sqrt{\frac{\alpha \ln t}{k_{a'}(t-1)}})$
7: **end if**
8: Inform the split layer index a to the IoT device and the edge server
9: Observe the service completion time $d_{a,t}$ and the service accuracy $acc_{a,t}$
10: Calculate the reward $r_{t,a}$ according to (4)
11: Update $\mathbf{v}$ and $\mathbf{k}$ according to (8) and (9), respectively
12: **end for**

서비스 처리 시간 $d_{a,t}$과 서비스 정확도 $acc_{a,t}$를 스플릿 컴퓨팅 서비스 컨트롤러에게 알려준다 (알고리즘1에서 8~9 번째 줄 동작과정). 컨트롤러는 획득한 스플릿 컴퓨팅 성능들을 기반으로 수식 (4)을 통해 서비스 품질 값을 계산한다 (알고리즘 1에서 10 번째 줄 동작과정). 끝으로, 선택한 레이어 a에 대한 선택 횟수 정보와 경험적으로 획득한 레이어 a에 대한 보상 값을 수식 (9) 과 (10)을 통해 갱신한다.

$$k_l(t) = \begin{cases} k_l(t-1)+1, if\ \ a=l \\ k_l(t-1), \qquad else \end{cases} \tag{8}$$

$$v_l(t) = \begin{cases} \frac{v(t-1)k_l(t-1)+r_{a,t}}{k_l(t-1)+1}, if\ \ a=l \\ v_l(t-1), \qquad else \end{cases} \tag{9}$$

Ⅳ. 실 험

본 장에서는 TESLS기법의 성능 평가 내용을 기술한다. 성능 평가를 하기 위한 실험 환경은 다음과 같다. 우선, 단말과 에지 서버의 평균 컴퓨팅 능력 (즉, $E[f_{l,l}]$와 $E[f_{E,l}]$)을 0.1 BFLOPS과 1 BFLOPS 로 설정하였다. 또한, 단말과 AP간의 평균 전송 속도는 ($E[b_{l,l}]$)는 10 Mbps이고 평균 전송 에러율은 ($E[E_{rate,l}]$) 는 10^{-3}로 설정하였다. IoT 서비스는 컨볼루션 레이어, 풀링 레이어와 덴스 레이어가 총 23 계층으로 이루어진 VGG-16 모델 기반의 Cifar-100 데이터[14]를 분류하는 서비스를 가정하였다[15]. 해당 심층 모델은 계층별로 각기 다른 요구 연산 량 (FLOPs)와 아웃풋 데이터 크기가 정의된다. 해당 서비스가 요구하는 최대 서비스 처리 시간 (d_{max})은 300 msec로 정의하였다. 또한, 서비스에 따른 파라미터인 α와 β는 0과 -1로 설정하였다[13]. 제안 기법의 성능을 비교하고자, 4가지의 비교기법을 다음과 같이 정의하였다. 우선, 스플릿 컴퓨팅 서비스를 적용하지 않고 단말에서 심층 모델을 모두 처리하는 MD 기법과 에지 서버에서 심층 모델을 모두 처리하는 ES 기법을 선정하였다. 또한, 스플릿 레이어를 임의로 선택하는 RAND 기법과 기존 연구들과 같이 심층 모델 추론 시간을 최소화하는 FAST 기법을 비교 기법들로 선정하였다.

4.1 온라인 학습 성능 평가

그림 5는 제안 기법의 온라인 학습에 대한 성능을 나타낸다. 온라인 학습 성능 평가를 위해 누적 후회 값 (cumulated regret)을 성능지표로 사용하였다. 누적 후회 값은 최적의 누적 보상 값과 학습 동안 발생하는

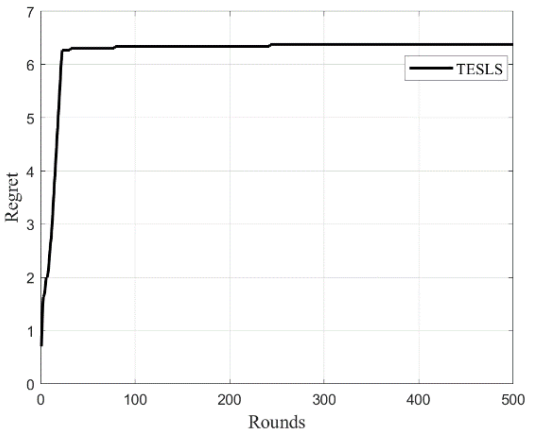


그림 5. 온라인 학습 성능 평가
Fig. 5. Regret values

누적 보상 값의 차이를 의미하고 수식 (10)과 같이 정의된다.

$$R(T) = \max_{i=1,\dots,L} E[\sum_{t=1}^{T} r_{i,t}] - E[\sum_{t=1}^{T} r_{\pi(t),t}] \tag{10}$$

수식 (10)의 앞의 수식 표현 ($\max_{i=1,\dots,L} E[\sum_{t=1}^{T} r_{i,t}]$)은 T 시점까지 최적의 평균 누적 보상 값을 의미하며, 뒤의 수식 표현 $E[\sum_{t=1}^{T} r_{\pi(t),t}]$은 제안 기법에서 학습되고 있는 정책 ($\pi(t)$)에 따른 누적 평균 보상 값을 나타낸다.

그림 5의 결과를 보면 누적 후회 값이 어느 시점까지 증가하다가 (T = 30) 더 이상 증가하지 않은 것을 확인할 수 있다. 이는 TESLS 기법이 시간이 지남에 따라 최적의 스플릿 레이어를 선택하기 때문이다. TESLS 기법에 의한 서비스 품질 값과 최적의 스플릿 레이어에 의한 서비스 품질 값의 차이가 클수록 *R(T)* 그래프의 기울기가 크고, 가까울수록 작다. 즉, *R(T)* 그래프의 기울기가 0 인 경우 최적의 스플릿 레이어를 환경과의 상호작용을 통해 학습을 완료한 것이고, 본 성능을 통해 IoT 서비스가 수행되는 동안 빠르게 최적의 스플릿 레이어를 학습하는 것을 확인하였다.

4.2 d_{max}에 따른 성능평가

그림 6은 IoT 서비스가 요구하는 최대 서비스 처리 시간 d_{max}에 따른 서비스 품질 값을 나타낸다. 서비스가 요구하는 최대 서비스 처리 시간이 짧을수록 실시간 서비스임을 나타낸다. 그림 6의 그래프를 통해 TESLS 기법이 항상 다른 기법보다 높은 서비스 품질 값을 제공하는 것을 확인할 수 있다. 이는 TESLS 기법이 서비스가 요구하는 최대 서비스 처리 시간을 고려하여, 해당

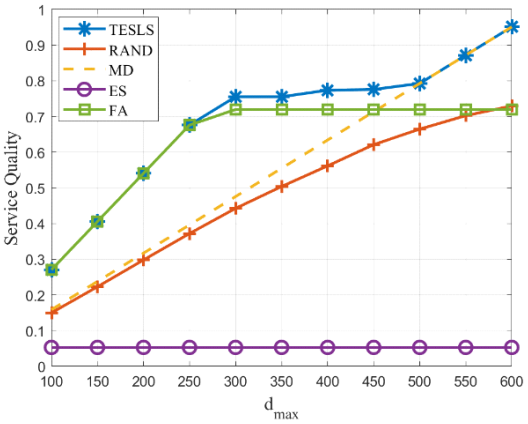


그림 6 요구 서비스 처리시간에 따른 성능
Fig. 6 The effect of d_{max}

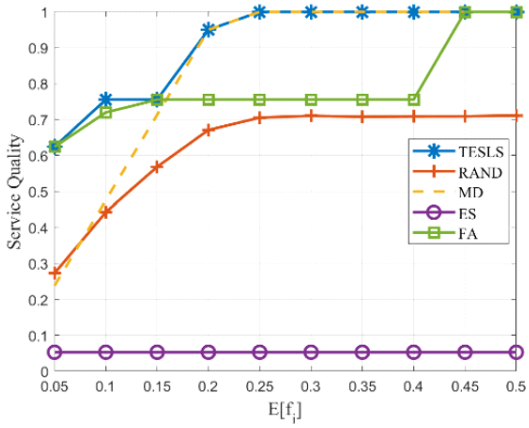


그림 7. $E[f_{i,l}]$에 따른 성능
Fig. 7. The effect of $E[f_{i,l}]$

시간내의 최대의 서비스 정확도를 높이기 위한 스플릿 레이어를 적응적으로 선택하기 때문이다. 그림 5를 통해 보면, d_{max} 값이 작은 경우, TESLS 기법은 실시간 서비스를 지원하기 위해 빠른 속도를 보장하는 FAST 기법과 비슷한 스플릿 레이어를 선택하기 때문에 FAST 기법 성능과 유사한 성능을 나타내는 것을 확인할 수 있다. 이와 반면에 d_{max} 값이 큰 경우 (즉, 비 실시간 서비스의 경우), 전송 에러율에 따른 서비스의 정확도 저하를 최소화하기 위해 많은 심층 레이어 들을 단말에서 처리하도록 스플릿 레이어를 선택하게 되는 것을 확인할 수 있다. 이러한 이유로 d_{max} 값이 큰 경우 경우 MD와 같은 서비스 품질을 보장하는 것을 확인할 수 있다. 에지 서버에서 모든 심층 레이어를 연산하는 ES의 경우 서비스 처리 시간을 짧지만 입력 데이터의 손실로 인한 서비스 정확도 저하가 크기 때문에 항상 가장 낮은 서비스 품질을 보여준다.

4.3 $E[f_{i,l}]$에 따른 성능평가

그림 7은 IoT 단말의 평균 컴퓨팅 능력에 따른 서비스 품질 값을 나타낸다. 그래프를 통해 제안하는 TESLS 기법이 항상 다른 기법보다 높은 서비스 품질 값을 제공하는 것을 확인할 수 있고, 단말의 평균 컴퓨팅 능력이 증가함에 따라 MD기법과 동일한 서비스 품질을 도출하는 것을 확인할 수 있다. 이는 IoT 단말 스스로의 심층 모델을 추론하여도 d_{max} 보다 빨리 추론을 완료 할 수 있기 때문이다. 이러한 상황에서는 통신 에러에 따른 추론 정확도가 감소되지 않게 하기 위해 (즉, 추론 정확도를 최대화 하기위해 TESLS기법은 적응적으로 단말의 많은 심층 레이어들을 처리하도록 스플릿 레이어를 결정한다. 최종적으로는 모든 레이어를 단말에서 처리하게 함으로써 서비스 품질 지수가 1의 값을 제공한다. 이와 반면에 ES의 경우 단말의 컴퓨팅 능력과 상관없이 항상 에지 서버로 오프로딩하기 때문에 늘 같은 서비스 품질을 제공하는 것을 확인 할 수 있다.

V. 결 론

본 논문에서는 통신 에러에 의한 스플릿 컴퓨팅 서비스의 영향력을 실험적으로 분석하였다. 실험적으로 분석한 결과를 토대로 IoT 서비스의 품질을 최대화하는 TESLS 기법을 제안하였으며, 다양한 실험을 통해 TESLS 기법의 온라인 학습 능력과 우수한 성능을 입증하였다. 향후에는 다수의 단말들이 혼재 되어있는 경우 통신 에러를 고려하여 평균 서비스 품질을 최대화하는 기법을 연구할 예정이다.

References

[1] "*The next hyper-connected experience for all*," [Online]. Available: https://news.samsung.com/global/samsungs-6g-white-paper-lays-out-the-companys-vision-for-the-next-generation-of-communications-technology

[2] 3GPP Technical Requirement (TR) 22.874, *Study on Traffic Characteristics and Performance Requirements for AI/ML Model Transfer*, v18.2.0, Dec. 2021.

[3] 3GPP Technical Requirement (TR) 23.700-80, *Study on 5G system support for AI/ML-based services*, v18.2.0, Dec. 2022.

[4] 3GPP Technical specification (TS) 23.501, *System architecture for the 5G System* (5GS), v18.3.0, Sep. 2023

[5] 3GPP Technical specification (TS) 23.502, *Procedures for the 5G System* (5GS), v18.3.0, Sep. 2023

[6] B. Kim and H. Ko, "Trends on the Split Computing," in *Proc. KICS Fall Conf.*, pp. 341-342, Nov. 2022.

[7] Y. Kang, et al., "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," in *Proc. ACM ASPLOS 2017*, Apr. 2017. (https://doi.org/10.1145/3037697.3037698)

[8] J. Chen, et al., "Accelerating DNN inference by edge-cloud collaboration," in Proc. *IEEE IPCCC 2021*, Oct. 2021. (https://doi.org/10.1109/ipccc51483.2021.9679434)

[9] S. Wang and X. Zhang, "NeuroMessenger: Towards error tolerant distributed machine learning over edge networks," in Proc. *IEEE INFOCOM 2022*, May 2022. (https://doi.org/10.1109/infocom48880.2022.9796695)

[10] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2-3, pp. 235-256, May 2002. (https://doi.org/10.1023/A:1013689704352)

[11] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *J. Mach. Learn. Res.*, vol. 3, pp. 397-422, Mar. 2003.

[12] D. Castelvecchi, "Can we open the black box of AI?," *Nature News,* vol. 538, no. 7623, pp. 20, Oct. 2016. (https://doi.org/10.1038/538020a)

[13] M. Tan, et al., "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF CVPR 2019*, Jun. 2019. (https://doi.org/10.1109/cvpr.2019.00293)

[14] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. Thesis, Dept. Comp. Sci., Univ. Toronto, Toronto, Canada, Apr. 2009.

[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR 2015*, May 2015.

이 재 욱 (Jaewook Lee)

2021년 8월 : 고려대학교 전기전자 공학 박사

2021년 9월~2023년 2월 : 한국전자통신 연구원 선임연구원

2023년 3월~현재 : 부경대학교 정보통신공학과 조교수

<관심분야> 6G 모바일 네트워크, Network for AI

[ORCID:0000-0003-0422-280X]

고 근 수 (Keunsoo Ko)

2023년 2월 : 고려대학교 전기전자 공학 박사

2023년 8월~현재 : 가톨릭대학교 인공지능학과 조교수

<관심분야> 컴퓨터 비전, 영상처리, 인공지능

[ORCID:0000-0003-0203-4530]

고 한 얼 (Haneul Ko)

2016년 8월 : 고려대학교 전기전자 공학 박사

2019년 9월~2022년 8월 : 고려대학교 컴퓨터융합소프트웨어학과 조교수

2022년 9월~현재 : 경희대학교 전자공학과 조교수

<관심분야> 6G 모바일 네트워크, Network for AI

[ORCID:0000-0002-9067-445X]