논문 21-46-08-07 The Journal of Korean Institute of Communications and Information Sciences '21-08 Vol.46 No.08
https://doi.org/10.7840/kics.2021.46.8.1262

엣지 컴퓨팅 기반의 딥러닝 서비스 모델을 위한 직교 특징벡터 추출 기법

이 종 관•

Orthogonal Feature Extraction Scheme for Edge Computing-Based Deep Learning Service

Jongkwan Lee•

요 약

본 논문은 컴퓨팅 및 네트워크 자원이 제한적인 단말에서 딥러닝 서비스를 효과적으로 활용하기 위한 직교 특징벡터 추출기법을 제안한다. 단말에서 딥러닝 서비스를 이용하기 위한 방안 중 하나는 분석하고자 하는 데이터를 원격 서버에 전송하여 서비스를 이용하는 것이다. 이러한 방안은 대용량 컴퓨팅 자원이 필요하지 않을 수 있다. 하지만 전송하는 데이터의 크기가 크다면 원격서버와의 통신을 위한 자원이 많이 필요하다. 또한 원격 서버가 물리적으로 멀리 이격되어 있는 경우에는 과도한 지연시간의 발생으로 원활한 서비스 이용이 제한된다. 따라서 딥러닝 서비스의 성능 저하 없이 단말에서 전송하는 데이터의 크기를 되도록 줄여야 한다. 제안하는 기법은 이러한 필요성을 충족하기 위해 단말에서 원본데이터로부터 저용량의 직교 특징벡터를 다수 추출하고 특징벡터들간의 직교성을 이용하여 전송되는 데이터의 크기를 줄인다. 또한 엣지 서버에서는 다수의 직교 특징벡터들을 이용한 앙상블 모델을 통해 딥러닝 서비스의 성능을 향상시킨다. Fashion-MNIST 데이터셋 대상으로 실험한 결과, 제안하는 기법이 전송데이터의 크기는 줄이면서 딥러닝 서비스의 성능을 향상시킬 수 있음을 확인하였다.

Key Words : Edge computing, Deep learning, Autoencoder, Orthogonal Feature Vector, End Devices

ABSTRACT

This paper proposes an orthogonal feature extraction scheme in the deep learning service model for end devices with limited computing and networking resources. One way to use the service is to use the service by transmitting the data to be analyzed to a powerful remote server. It may not require a large number of resources for end devices. However, if the size of the transmitted data is large, many resources are needed for communication. Besides, when the remote server is located far from the devices, it is unavailable due to excessive latency. Therefore, it is necessary to reduce the data size without deteriorating the deep learning performance. In the proposed scheme, the device extracts many low-capacity orthogonal feature vectors, and the edge server improves the deep learning performance through an ensemble model using the orthogonal feature vectors. Experimenting on the Fashion-MNIST dataset confirmed that the proposed scheme improves the deep learning service's performance while reducing the size of transmitted data.

※ 이 논문은 한국연구재단(No. NRF-2019R1G1A100303013) 및 육군사관학교 화랑대연구소의 지원으로 수행되었음
• First Author : Korea Military Academy Department of Computer Science, jklee64@kma.ac.kr, 정회원
논문번호 : 202103-054-B-RN, Received March 9, 2021; Revised April 24, 2021; Accepted April 26, 2021

Ⅰ. 서 론

4차 산업혁명의 핵심 기술 중 하나로 항상 언급되는 인공지능 기술은 이제 미래의 기술이 아닌 현재의 기술로서 우리의 일상 속에서 조금씩 구현되고 있으며, 그 적용 분야는 점차 확대되고 있다. 인공지능을 구현하는 많은 방법 중 가장 우수한 성능을 나타내는 기술은 딥러닝이다. 딥러닝은 은닉계층이 다수인 인공신경망으로 모델 학습을 위해 많은 데이터가 필요하다. 또한 해결하고자 하는 문제의 난이도가 증가함에 따라 딥러닝 모델의 복잡도도 급격히 커지고 있는 추세이다[1-4].

단말에서 딥러닝을 수행하기 위한 방안으로는 단말의 제한적 자원을 고려하여 경량화된 딥러닝 모델을 설계하거나 모델의 파라미터 양자화(quantization)[5,6], 파라미터 가지치기(pruning)[7,8], 지식증류(knowledge distillation)[9,10] 등의 기법으로 모델을 압축하는 방법이 있다. 또는 단말에 적합한 딥러닝 전용 하드웨어를 사용할 수도 있다. 이와 같은 방법으로 간단한 딥러닝 서비스는 가능하다. 하지만 지속적인 전원 공급이 제한되는 모바일 기기, 저전력으로 동작해야 하는 센서, 가격 대비 성능 측면에서 고가의 하드웨어 탑재가 어려운 단말 등에 딥러닝 모델을 직접 구현하는 것은 적절하지 않다[11].

이러한 문제를 해결하기 위한 대안이 엣지 컴퓨팅, 클라우드 컴퓨팅 또는 엣지-클라우드 컴퓨팅을 이용하는 것이다. 딥러닝 계산작업의 일부 또는 전부를 엣지 또는 클라우드에 분산하는 것이다. 단말은 분석 대상 데이터를 엣지(또는 클라우드)에 전송하고 딥러닝 모델학습과 모델추론은 엣지 또는 클라우드에서 수행되며, 추론 결과는 다시 단말에게 전송된다. 이와 같은 형태의 딥러닝 서비스 구현은 단말이 고성능의 하드웨어를 구비할 필요가 없다는 큰 장점이 있다. 하지만 클라우드를 이용하는 경우에는 적지 않은 네트워크 지연시간이 발생할 수밖에 없기 때문에 실시간성 서비스에는 적합하지 않다. 뿐만 아니라 클라우드에 접속하기 위해 공공(public) 네트워크를 경유한다면 민감한 정보전송에 대한 보안 이슈도 해결해야할 과제이다.

한편, 엣지는 단말 인근에 위치한 서버이기 때문에 클라우드에 비해 네트워크 지연시간이나 데이터 보안 이슈 측면에서 유리하다. 하지만 엣지까지의 전송로가 무선이라면 단말의 이동, 무선채널의 불안정성, 인접 단말의 채널 사용 등에 따라 전송시간이 가변적일 수 있다. 또한 전송되는 데이터가 고화질 이미지, 고음질 음향신호 또는 동영상과 같이 데이터 자체의 용량이 크다면 전송시간이 증가할 수밖에 없다. 따라서 엣지 기반의 딥러닝 서비스라 하더라도 단말에서 엣지까지의 전송시간을 가급적 최소화하는 것이 필요하다.

엣지로 전송되는 데이터의 크기를 줄이기 위한 다양한 연구가 있다. Glimpse는 모든 딥러닝 계산을 엣지 서버가 수행하는 기법으로 단말이 원본 데이터를 인근 엣지 서버에 전송하되 이전 프레임에 비해서 변경된 사항만을 전송하여 전송시간을 줄였다[12]. Liu 등은 음식 인식 시스템을 구현하는데 있어 흐릿한 이미지는 전송하지 않고 인식해야 하는 대상체 중심으로 이미지를 잘라내는 전처리를 수행한 후 엣지에 데이터를 전송하여 전송시간을 줄였다[13]. 이들 연구는 원본 데이터의 불필요한 전송 자체를 줄이거나 원본 데이터의 일부만을 전송하는 것으로 고차원의 원본 데이터에 포함되어 있는 중복성분(redundancy)을 제거하지는 않는다. 물론 딥러닝 모델은 중복성분이 제거된 특징을 추출하는 과정이 포함되어 있다. 하지만 데이터 전송 전에 중복성분이 제거되면 보다 작은 크기로 데이터를 전송할 수 있다.

본 논문에서는 엣지 컴퓨팅 기반의 딥러닝 서비스에서 단말이 데이터를 전송할 때 발생하는 전송지연시간을 줄이기 위해 고차원의 데이터에서 저차원의 특징벡터들을 추출하여 엣지로 전송하는 방안을 제안한다. 논문의 구성은 다음과 같다. 2장에서 논문에서 고려하는 딥러닝 서비스 모델과 가정사항에 대해서 살펴본다. 3장에서는 제안하는 특징벡터 추출 및 전송 기법에 대해서 상세한 다루고 4장에서 제안하는 기법의 성능을 실험을 통해 분석한다. 마지막으로 5장에서 결론 및 향후계획에 대해 논한다.

Ⅱ. 딥러닝 서비스 모델 및 가정 사항

본 논문에서 고려하는 딥러닝 서비스 모델 아키텍처는 그림 1과 같이 단말-엣지-클라우드 계층으로 구성되며 주요 가정사항은 다음과 같다.

단말 계층은 사용자 또는 수집하고자 하는 데이터와 직접적인 접점이 되는 단말들로 구성된다. 단말 계층에 속하는 기기들은 한정된 자원을 보유하고 있어 대규모 딥러닝 모델의 학습 또는 추론이 제한적이다.

엣지 계층은 딥러닝 모델 추론을 수행하는 서버들로 구성된다. 엣지 서버는 추론을 신속히 수행할 수 있을 정도의 연산파워를 갖고 있지만 대용량의 데이터를 처리하여 딥러닝 모델을 학습하거나 갱신하기에는 연산파워가 부족하다. 한편, 엣지 서버는 단말 계

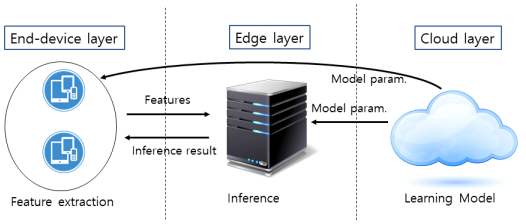

그림 1. 엣지-클라우드 컴퓨팅 기반 딥러닝 서비스 모델
Fig. 1. Deep learning service model based on egde-cloud computing

층과 물리적으로 근접해 있어 단말과의 네트워크 비용이 저렴하다.

마지막으로 클라우드 계층은 딥러닝 모델을 학습하는 기능을 수행하는 서버로 구성된다. 클라우드 서버는 엣지 서버에 비해 월등한 연산파워를 갖고 있어 딥러닝 모델의 학습, 추론 등이 가능하다. 하지만 단말 계층과 물리적으로 멀리 이격되어 있어 단말과의 통신을 위한 네트워크 비용이 크다.

한편, 딥러닝 서비스는 다음과 같은 절차에 의해서 제공된다. 우선, 단말은 분석하고자 하는 수집된 데이터의 특징을 추출하고 엣지 서버에 이를 전송한다. 엣지 서버는 수신된 특징을 이용하여 딥러닝 추론을 수행하고 추론 결과를 다시 단말에 전송한다. 한편, 클라우드 서버는 별도로 수집되는 대량의 데이터를 이용하여 딥러닝 모델을 학습하며, 주기적으로 모델을 최신화하여 딥러닝 모델을 엣지 서버 또는 단말에 배포한다.

Ⅲ. 제안하는 딥러닝 서비스 모델

본 장에서는 제안하는 딥러닝 서비스 모델에 대해 구체적으로 살펴본다. 표 1은 제안하는 시스템 모델을 설명하기 위한 주요 기호와 의미이다.

표 1. 주요 기호와 의미
Table 1. Notation

Symbol	Meaning
AE_i	i^{th} autoencoder model in feature extraction model
E_i	encoder part in AE_i
D_i	decoder part in AE_i
OF_i	i^{th} feature vector
$OF_i(k)$	k^{th} element in OF_i
$CS(A,B)$	cosine similarity between vector A and B
n	number of autoencoder in feature extraction model
l	length of feature vector

3.1 특징벡터 추출 모델

오토인코더는 입력데이터와 출력데이터가 같아지도록 학습하는 비지도학습 신경망 모델로 그림 2와 같이 인코더와 디코더 부분으로 구성된다. 인코더 출력의 차원을 입력의 차원보다 작게 하여 데이터 압축, 차원 축소, 특징 추출 등의 목적으로 사용될 수 있다[14]. 제안하는 서비스 모델에서는 단말에서 n개의 오토인코더를 이용하여 하나의 입력 데이터에 대해 상호 직교하는 n개의 특징벡터를 추출한다.

제안하는 딥러닝 서비스 모델에서 특징벡터 추출을 위한 모델의 구조는 그림 3과 같다.

입력데이터 x는 n개의 단일 오토인코더에 동일하게 입력되며, 전체 손실함수는 각 오토인코더의 재현율 오차와 각 인코더의 출력벡터들 간의 직교성에 따라 결정된다. 즉, 재현율 오차가 작아지는 동시에 오토인코더의 출력벡터들이 서로 직교하는 방향으로 파라미터들이 최적화된다. 손실함수는 식 (1)과 같다.

$$L_\lambda(\omega,x)=\frac{1}{n}\parallel D_i(E_i(x))-x\parallel^2 +\lambda\sum_{i=1}^{n}\sum_{j>i}^{n}\left|CS(E_i(x),E_j(x))\right| \quad (1)$$

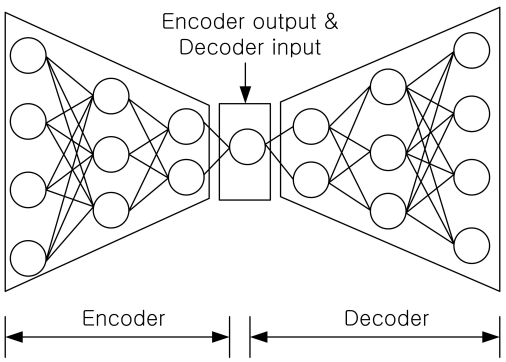

그림 2. 오토인코더 모델의 구조
Fig. 2. Structure of autoencoder model

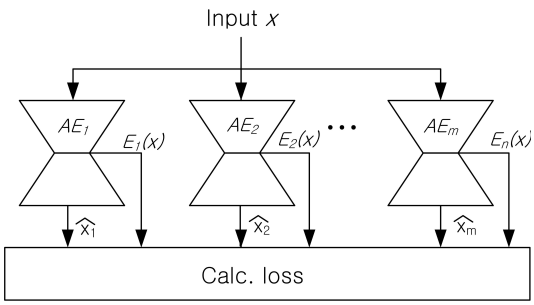

그림 3. 특징 추출 모델 구조
Fig. 3. Structure of model for feature extraction

경사하강법으로 손실함수를 최소화하는 파라미터 ω를 학습한다. 식 (1)에서 λ는 하이퍼 파라미터로 손실함수에서 인코더 출력벡터들의 직교성 정도를 반영하는 비율을 조정한다. λ=0 이면 재현율 오차는 최소화되지만 인코더 출력 벡터들간의 직교성은 학습시 고려되지 않는다. λ가 클수록 학습시 출력벡터들간의 직교성이 우선적으로 고려된다.

한편, 출력벡터들간의 직교성은 코사인 유사도를 이용하여 계량화한다. 벡터 A, B에 대한 코사인 유사도는 식 (2)와 같이 계산된다.

$$CS(A,B)=\frac{A\cdot B}{\|A\|\ \|B\|}=\frac{\sum_{i=1}^{n}A_iB_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2}\sqrt{\sum_{i=1}^{n}(B_i)^2}} \quad (2)$$

코사인 유사도는 -1과 1사이의 값을 갖는데 두 벡터의 방향이 완전히 같은 경우 1, 90°의 각을 이루는 경우 0, 완전히 반대 방향인 경우 -1의 값을 갖는다.

제안한 손실함수에서는 코사인 유사도의 절대값이 사용된다. 따라서 손실함수가 최소화되기 위해서는 코사인 유사도가 0이 되어야 한다. 즉, 인코더들의 출력벡터들간의 코사인 유사도가 0이 되도록 학습된다. 이것은 각 출력벡터들이 직교한다는 의미이다. 특징벡터 추출 모델 학습이 종료된 이후에 단말은 오토인코더들의 인코더 부분들만 사용한다.

3.2 특징벡터 추출 및 전송

단말에서 추출되는 n개의 특징벡터는 서로 직교한다. 다시 말해, i번째 특징벡터의 k번째 원소가 0이 아니라면, 나머지 특징벡터에서의 k번째 원소는 모두 0이다. 특징벡터는 평균적으로 $\left(l-\left\lfloor\frac{l}{n}\right\rfloor\right)$개의 0을 포함하고 있으며, 각 원소는 16비트의 길이를 갖는다. 따라서 모든 특징벡터를 엣지에 전송하는 경우의 전송데이터 크기는 $16\times n\times l$가 된다. 그런데 각 특징벡터에는 0이 다수 포함되어 있기 때문에 0을 제외한 원소들의 위치정보를 l 비트로 표현할 수 있다. 즉, 위치정보는 특징벡터의 원소값이 0이 아닌 위치는 1로 0인 위치는 0으로 매핑한 비트열이다. 각 특징벡터의 위치정보와 특징벡터의 원소 중 0이 아닌 원소들만을 결합하여 전송한다. 또한 마지막 특징벡터의 위치정보는 나머지 위치정보들을 통해 추정될 수 있다. 따라서 전송데이터의 크기는 평균적으로 식(3) 같이 계산된다.

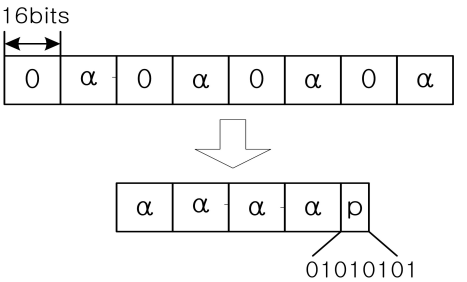

그림 4. 전송데이터 크기 감소의 예
Fig. 4. Example of reduction of data size

$$DS\simeq 16n\left\lfloor\frac{l}{n}\right\rfloor+(n-1)l \quad (3)$$

그림 4는 특징벡터의 크기를 감소시키는 개념도를 나타낸다. 그림은 16비트 부동 소수점 데이터 8개를 0이 아닌 값들과 위치정보의 결합을 통해 전송데이터의 크기를 감소시키는 예이다.

3.3 앙상블 분류 모델

n개의 특징벡터를 수신한 엣지에서 입력데이터에 대한 분류가 수행된다. 동일한 데이터에 대해 n개의 특징벡터가 있기 때문에 제안하는 서비스 모델은 앙상블 모델을 사용한다. 앙상블 모델은 다수의 분류 모델의 출력들을 적절히 조합하여 최종적으로 분류 결과를 결정한다. 이를 통해 분류 결과에 대한 분산을 줄이고 과적합을 방지할 수 있다[15].

그림 5는 제안하는 서비스 모델에서 사용되는 앙상블 분류 모델을 나타낸다. 단말로부터 수신한 n개의 특징벡터는 각각 분류기에 개별적으로 입력되고 분류기의 출력들을 앙상블 처리하여 최종 분류결과를 결정한다. 이후 엣지는 분류결과를 단말에 전송한다.

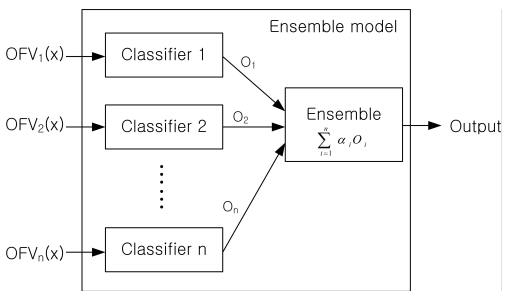

그림 5. 앙상블 분류 모델
Fig. 5. Ensemble model for classification

3.4 모델 학습

제안하는 서비스 모델에 적용되는 딥러닝 모델은 단말에서 사용하는 특징벡터 추출 모델과 엣지에서 사용하는 앙상블 분류모델이 있다. 두 모델의 학습은 모두 클라우드에서 수행된다. 딥러닝 학습에는 큰 계산량이 요구되기 때문에 컴퓨팅 자원이 충분한 클라우드에서 학습이 수행되고 신규 모델 또는 갱신 모델에 대한 파라미터들을 필요시 클라우드에서 단말 또는 엣지로 전송한다.

Ⅳ. 성능 분석

4.1 실험 환경

4.1.1 데이터 셋

성능분석을 위한 실험에 사용된 데이터셋은 셔츠, 바지 등 10가지 종류의 의류 이미지로 구성된 Fashion-MNIST이다. Fashion-MNIST는 6만장의 학습데이터와 1만장의 테스트데이터로 구성되었으며, 각 데이터는 28×28 픽셀 크기의 그레이스케일 이미지이다[16].

4.1.2 특징벡터 추출 모델

특징벡터 추출 모델의 구성요소로 사용되는 오토인코더는 모두 동일한 구조로 표 2와 같다. 오토인코더는 입력계층, 은닉계층, 출력계층 등 3개의 계층으로 구성된다. 각 오토인코더의 은닉계층의 출력이 특징벡터가 된다. 따라서 은닉계층의 뉴런의 수(l)는 특징벡터의 크기에 따라 결정된다.

단말은 오토인코더 모델의 인코더 부분 즉 입력계층과 은닉계층만 사용하여 특징벡터들을 추출한다. 전체 모델은 클라우드 계층에서 학습시 사용된다.

표 2. 특징 추출 모델 구조
Table 2. Structure of model for feature extraction

	# of neuron	activation func.
input layer	786	-
hidden layer	l	Relu
output layer	786	Sigmoid

4.1.3 앙상블 분류 모델

앙상블 분류 모델을 구성하는 개별 분류기는 완전연결 신경망 모델이다. 개별 분류기의 수는 입력되는 특징벡터의 수와 같다. 그리고 각 분류기는 모두 동일한 구조로 표 3과 같다. 개별 분류모델은 입력계층, 2개의 은닉계층, 출력계층 등 4개의 계층으로 구성된다. 입력계층의 뉴런 수(l)는 입력되는 특징벡터의 크기에 따라 결정된다.

표 3. 분류기 모델 구조
Table 3. Structure of classification model

	# of neuron	activation func.
input layer	l	-
hidden layer 1	64	Relu
hidden layer 2	64	Relu
output layer	10	Sigmoid

4.2 실험 결과

4.2.1 특징벡터의 직교성 분석

특징벡터들의 상호 직교성의 정도를 분석하기 위해 특징벡터들간의 평균 코사인 유사도를 이용하여 아래와 같이 직교성 지표(OI: Orthogonality Index)를 정의한다.

$$OI = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=i+1}^{n} CS(FV_i, FV_j) \tag{3}$$

직교성은 1만개의 테스트데이터에 대한 특징벡터들의 평균값으로 0에 가까울수록 직교성이 높고 1에 가까울수록 직교성이 낮은 것이다. 표 4는 λ가 0일 때 추출되는 특징벡터의 수(즉, n)의 값에 따른 직교성을 나타낸다. 이때 특징벡터의 길이(l)은 512이다. 표에서 알 수 있는 바와 같이 λ가 0일 때 특징벡터들은 서로 직교하지 않으며 n과도 무관하다.

한편, 그림 6은 특징벡터의 길이가 512일 때, λ 값에 따른 특징벡터들의 직교성을 나타낸다. y축의 단위가 10^{-5}임에 유의한다. 그래프에서 알 수 있는 것과 같이 λ가 커질수록 평균 직교성이 0에 더욱 가까워진다. 그런데 λ가 1이상이면 이미 평균 직교성이 0에 매우 가깝게 된다. 즉, 충분한 직교성이 보장된다. 이것은 특징벡터들간의 직교성 정도가 하이퍼 파라미터인 λ 값에 영향을 받지만 λ의 선택이 쉽다는 것을 의미한다. 한편, 추출하는 특징벡터의 수가 증가할수록 평균 직교성이 0에 가깝다.

표 4. λ=0일 때 서로 다른 n의 값에 따른 직교성
Table 4. Orthogonality for different values of n in case of λ=0

n	2	3	4	5
OI	0.715	0.717	0.722	0.716

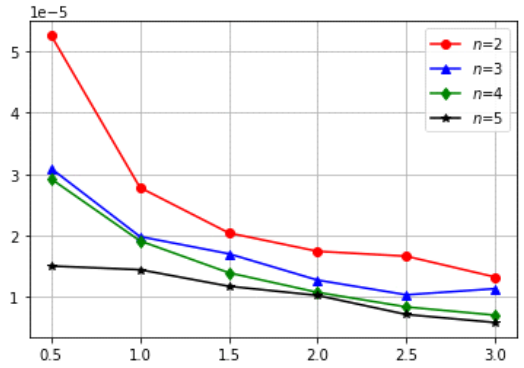

그림 6. 다양한 λ값에 따른 직교성 지표 (l=512)
Fig. 6. Orthogonality Index for different values of λ (l=512)

결론적으로 λ가 클수록 그리고 추출되는 특징벡터의 수가 많을수록 특징벡터들은 서로 직교하는 특성이 강하다.

4.2.2 F1 점수 및 전송데이터 크기 분석

제안하는 시스템의 성능을 평가하기 위해 F1 점수와 전송데이터 크기를 분석한다. 대표적인 특징 추출 기법인 PCA 특징벡터와 단일 오토인코더 모델을 이용한 특징벡터들을 제안하는 시스템의 특징벡터들과 비교하였다.

각 특징벡터에 사용된 분류기의 모델 구조는 모두 동일하며 각 분류기는 PCA에 의해 추출된 특징벡터, 단일 오토인코더 모델에 의해서 추출된 특징벡터, 제안한 시스템에 의해서 추출된 특징벡터들로 각각 학습되었다. 또한 학습에 사용된 데이터는 모두 동일하다.

표 5는 각 특징벡터에 의한 분류기의 F1 점수와 전송데이터의 크기를 나타낸다. 원본데이터를 분류기에 직접 입력하는 경우의 F1 점수는 0.8814이고, 전송데이터의 크기는 6,272 비트이다. PCA 특징벡터는 특징벡터의 크기가 64까지 증가할 때 분류기의 F1 점수가 증가하지만 일정 수준 이상으로 특징벡터 크기가 증가하면 오히려 F1 점수가 저하된다. 이는 특징벡터의 크기가 증가할수록 학습에 필요한 데이터의 수가 증가하기 때문이다[17]. 또한 특징벡터의 크기가 증가할수록 추가되는 PCA 요소들은 원본 데이터를 표현하는 특징으로서의 가치가 점점 낮아진다. 따라서 특징벡터의 크기가 증가하더라도 F1 점수가 증가하지 않고 오히려 감소하게 된다. 한편, PCA 특징벡터를 사용했을 때의 F1 점수는 원본데이터를 이용한 모델의 F1 점수보다 모두 낮다.

단일 오토인코더에 의해 추출된 특징은 특징벡터의 크기가 증가할수록 분류기의 F1 점수가 증가하지만 일정 수준 이상으로 크기가 증가하면 F1 점수는 감소하기 시작한다. 이는 PCA 특징벡터의 경우와 마찬가지로 특징벡터의 크기가 증가할수록 학습에 필요한 데이터의 수가 많아지기 때문이다. 한편, *l*=32인 경우를 제외하고 PCA 특징벡터에 비해 단일 오토인코더에 의해 추출된 특징벡터를 사용했을 때 F1 점수가 높았지만, 원본데이터를 이용한 모델에 비하면 F1 점수가 낮게 나타났다.

다른 특징벡터들과 마찬가지로 OFV의 경우에도 특징벡터의 크기가 증가할수록 F1 점수가 일반적으로 향상되지만 일정 수준 이상으로 크기가 증가하면 오히려 F1 점수가 감소하기 시작한다. 또한 특징벡터의 크기가 증가할수록 전송해야 하는 데이터의 크기도 비례적으로 증가한다. 한편, 원본데이터의 크기에 비해 전송데이터의 크기가 큰 경우는 제안하는 시스템의 목적에 부합되지 않으므로 F1 점수가 높다 하더라도 무의미하다. 실험결과에서 보는 바와 같이 적절한 *n*과 *l*을 선택한다면(*n*=3, *l*=64), 원본데이터를 전송하여 분류하는 모델에 비해 F1 점수의 저하 없이 전송데이터의 크기를 최대 81% 감소시킬 수 있다. 또한 *l*

표 5. PCA, 단일 AE 모델, 제안한 시스템으로 추출된 특징벡터들의 F1 점수와 전송데이터 크기 비교
Table 5. Comparison of F1 score and data size between different features from PCA, Single AE, and proposed system

l	raw data		PCA		Single AE		OFV(n=2)		OFV(n=3)		OFV(n=4)	
	F1	Data size	F1	Data size	F1	Data size	F1	Data size	F1	Data size	F1	Data size
32	-	-	0.8682	512	0.8632	512	0.8689	544	0.8618	576	0.8574	608
64	-	-	0.8719	1,024	0.8738	1,024	0.8805	1,088	0.8825	1,152	0.8811	1,216
128	-	-	0.8688	2,048	0.8854	2,048	0.8865	2,176	0.8915	2,304	0.8849	2,432
256	-	-	0.8555	4,096	0.8787	4,096	0.8873	4,352	0.8936	4,608	0.8909	4,864
512	-	-	0.8389	8,192	0.8743	8,192	0.8898	8,704	0.8982	9,216	0.8886	9,728
786	0.8814	6,272	-	-	-	-	-	-	-	-	-	-

의 크기를 증가시키거나 n=4를 선택하면 전송데이터의 크기를 다소 증가시키지만 F1 점수를 향상시킬 수 있다. F1 점수가 원본 데이터를 이용한 것에 비해 향상되는 것은 다수의 직교 특징벡터들을 이용한 앙상블 효과 때문으로 분석된다.

V. 결론 및 향후 연구

본 논문에서 엣지 컴퓨팅 기반의 딥러닝 서비스 모델과 이를 위한 특징벡터 추출 기법을 제안하였다. 컴퓨팅 및 네트워크 자원이 충분하지 않은 단말에서 딥러닝 서비스를 효과적으로 활용하기 위해서는 엣지 컴퓨팅을 이용하고 단말에서 엣지 서버로 전송하는 데이터의 크기를 줄여야 한다. 따라서 딥러닝 모델의 성능 저하를 최소화한 상태에서 원본 데이터로부터 저용량의 특징을 추출하고 이를 효과적으로 전송할 필요가 있다.

제안하는 특징추출 기법은 원본데이터로부터 여러 개의 특징벡터를 추출하고 특징벡터들 간의 상관관계를 이용하여 특징벡터들의 전송크기를 줄인다. 그리고 엣지 서버에서는 단말로부터 수신한 다수의 특징벡터들을 이용한 앙상블 모델을 통해 모델의 성능을 향상시킨다. Fashion-MNIST 데이터셋을 대상으로 제안한 기법을 적용한 결과 적절한 특징벡터 크기와 특징벡터의 수를 선택했을 때 전송데이터의 크기는 줄이면서 분류기의 성능은 향상시킬 수 있음을 확인하였다.

향후계획으로 단말에서 전송해야 하는 특징을 추출하는데 요구되는 컴퓨팅 자원의 양을 최적화하기 위한 연구를 추가적으로 수행할 예정이다.

References

[1] X. Du, et al., "Overview of deep learning," *IEEE 31st Youth Academic Annual Conf. Chinese Assoc. of Automat.* (YAC), pp. 159-164, 2016.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint, arXiv: 1409.1556, 2014.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, and A. Rebinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. CVPR*, pp. 1-9, 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv preprint, arXiv: 1512.03385, 2015.

[5] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," arXiv:1510.00149, 2015.

[6] L. Lai and N. Suda, "Enabling deep learning at the IoT edge," in *Proc. ICCAD*, pp. 1-6, 2018.

[7] S. Han, et al., "ESE: Efficient speech recognition engine with sparse LSTM on FPGA," in *Proc. ACM/SIGDA Int. Symp. FPGA*, pp. 75-84, 2017.

[8] S. Srinivas and R. Venkatesh Babu, "Data-free parameter pruning for deep neural networks," arXiv preprint arXiv:1507.06149, 2015.

[9] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint arXiv:1503.02531, 2015.

[10] S. Yao, Y. Zhao, Z. Aston, L. Su, and T. Abdelzaher, "On-demand deep model compression for mobile devices: A usage-driven model selection framework," in *Proc. MobiSys*, pp. 389-400, 2018.

[11] J. Chen and X. Ran, "Deep learning with edge computing: A review," in *Proc. IEEE*, vol. 107, no. 8, pp. 1655-1674, 2019.

[12] T. Y.-H. Chen, et al., "Glimpse: Continuous, real-time object recognition on mobile devices," in *Proc. 13th ACM Conf. Embedded Networked Sensor Syst.*, 2015.

[13] C. Liu, et al., "A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure," *IEEE Trans. Serv. Comput.*, vol. 11, no. 2, pp. 249-261, 2017.

[14] D. Charte, et al., "A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines," *Info. Fusion*, vol. 44, pp. 78-96, 2018.

[15] X. Dong, et al., "A survey on ensemble

learning," *Frontiers of Comput. Sci.*, vol. 14, no. 2, pp. 241-258, 2020.

[16] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," arXiv preprint arXiv:1708.07747, 2017.

[17] G. Hughes, "On the mean accuracy of statistical pattern recognizers" *IEEE Trans. Info. Theory*, vol. 14, no.1, pp. 55-63, 1968.

이 종 관 (Jongkwan Lee)

2000년 2월 : 육군사관학교 전자공학과 학사

2004년 2월 : 한국과학기술원 전기 및 전자공학과 석사

2014년 2월 : 아주대학교 NCW 공학과 박사

2017년 12월~현재 : 육군사관학교 컴퓨터과학과 부교수

<관심분야> 인공지능, 사이버전, 전술네트워크

[ORCID:0000-0003-2195-2417]