KNOM Review '22-02 Vol.25 No.02
https://doi.org/10.22670/knom.2022.25.2.40

경량화 데이터와 딥러닝 모델을 적용한 효율적인 네트워크 트래픽 분류 방법

신 창 의◆, 박 지 태*, 백 의 준*, 최 정 우*, 김 명 섭°

Efficient Network Traffic Classification Method Using Lightweight Data and Deep Learning Model

Chang-Yui Shin◆, Jee-Tae Park*, Ui-Jun Baek*, Jung-woo Choi*, Myung-Sup Kim°

요 약

트래픽 분류는 컴퓨터 네트워크 영역에서 서비스관리 및 보안 등의 분야에서 그 역할이 점점 더 중요해지고 있다. 초기에는 포트넘버, DPI, 통계정보 등을 활용해 트래픽 분류가 가능했다. 그러나 정보보호 측면에서 트래픽의 페이로드가 암호화되면서 분류가 제한되었지만 머신러닝기법이 추가 활용되면서 문제점이 해결됐다. 이후 딥러닝 모델들이 활용되고 성능은 향상되었으나, 많은 변수를 입력으로 넣어도 트래픽 분류가 가능해 짐에 따라 모델과 데이터가 점점 무거워져서 자원과 시간이 많이 소모되었다. 부담스럽고 성가시게 된 모델과 데이터 경량화 활용에 본 연구의 목적을 두고, KD(Knowledge distillation) 기법을 바탕으로 BERT가 경량화된 DistilBERT를 선정했고, 경량화한 데이터를 적용했다. 첫 번째 패킷 1개 중 앞 100bytes 크기의 입력데이터(패킷 단위)와 이러한 5개의 패킷이 연결된 입력데이터(플로우 단위)로 정확도 / F1-score가 각각 0.9707 / 0.9731 과 0.9703 / 0.9706로 매우 우수한 성능을 보였다.

Key Words : Encrypted Network Traffic Classification, BERT, Knowledge Distillation, Efficiency

ABSTRACT

Traffic classification is becoming more and more important in areas such as service management and security in the area of computer networks. In the early days, it was possible to classify traffic using port numbers, DPI, and statistical information. However, in terms of information protection, classification was limited as the traffic payload was encrypted, but the problem was solved by additionally using machine learning techniques. Since then, deep learning models have been used and performance has improved, but as traffic classification became possible even with a large number of features as inputs, models and data became increasingly heavy like the front and back of a coin, consuming a lot of resources and time. With the purpose of this study to lighten the burdensome and cumbersome model and data, DistilBERT selected the lightweight BERT based on the previously presented KD (Knowledge distillation) research and applied the lightened data. With a 100bytes input of the first packet(packet unit) and an input(flow unit) of these 5 packets, the accuracy / F1-score was 0.9707 / 0.9731 and 0.9703 / 0.9706, respectively, showing very good performance.

※ 이 논문은 2021년도 교육부의 재원으로 한국연구재단의 지원을 받아 수행된 지자체-대학 협력기반 지역혁신 사업의 결과(2021RIS-004)이고, 2020년도 산업통상자원부 및 한국산업기술평가관리원(KEIT) 연구비 지원에 의한 연구(No. 20008902, IT비용 최소화를 위한 5채널 탐지기술 기반 SaaS SW Management Platform(SMP) 개발) 결과다.
◆ First Author : Defense Agency for Technology and Quality, superego99@gmail.com
° Corresponding Author : Korea University, Department of Computer Convergence Software, tmskim@korea.ac.kr
* Korea University, Department of Computer Convergence Software, {pjj5846, pb1069, choigoya97}@korea.ac.kr
논문번호 : KNOM2022-02-08, Received December 2, 2022; Revised December 10, 2022; Accepted December 16, 2022

Ⅰ. 서 론

네트워크에 연결된 퍼스널 디바이스, 다양한 서비스용 장비, 클라우드 서버 등이 인터넷을 통해 상호 연결되어 데이터를 공유하여 SNS, 헬스, 교통, 에너지관리 등의 서비스 산업들이 경계구분 없이 급격히 성장하고 있다. 이러한 서비스들의 QoS, 지능네트워크 운영, 관리 및 보안 등을 보장하기 위한 좋은 해결책이 네트워크 트래픽 분류 및 식별인데, 이것은 컴퓨터 및 네트워크 기술의 발전과 함께 지속적으로 계속 연구되고 있다.

트래픽 클래스 분류는 통상 트래픽 타입, 프로토콜, 애플리케이션 등의 3가지[1]에 따라 이루어진다. 초기 단계의 트래픽 분류는 IANA에 등록된 전송계층의 포트번호에 크게 의존하는 것이 일반적이다. 물론 포트번호 정보 1가지로만 분류하는 것이 아니고, 패킷도착시간, 패킷길이 및 플로우 분포 등의 통계적 정보들도 부가적으로 사용되는 연구들이 줄을 이었다. 그 다음단계에서 진행되는 트래픽 분류의 흐름은 페이로드에서 특정 패턴이나 키워드를 찾아서 동일한 패킷들 간의 연관성을 통해서 분류하는 것[2]으로써 포트번호만 가지고 분류하는 것보다 높은 정확도를 보였다. 그도 그럴 것이 잘 알려진 포트를 사용하지 않는 애플리케이션들이 등장하게 된 탓이다.

그러나 이러한 Deep Packet Inspection(DPI) 기법도 페이로드가 암호화되면서 지속적으로 트래픽분류에 적용하기에 제한되었다. 그 다음 단계에서 대표적으로 등장한 것이 머신러닝기법[3]을 활용한 트래픽 분류였다. 트래픽 플로우의 처음 몇 패킷을 사용하고 Simple K-means에 기반해 서로 다른 TCP 기반 애플리케이션 프로그램을 분류하는 머신러닝기법의 접근[4]이 있었다. 웹 및 P2P 트래픽을 식별하기 위해 K-means를 기반으로 누락된 통계를 추정할 수 있는 머신러닝 기법의 접근[5]도 그 뒤를 이었다. 양방향 플로우 기반이든 단방향 플로우 기반이든 의미 있는 특징을 추출하여 K-Nearest Neighbors, Decision Tree, Naive Bayes 및 Support Vector Machine(SVM) 등의 머신러닝 기법의 알고리즘을 활용한 트래픽분류가 대세가 된 것이다.

그 이후에 Computer Vision(CV) 및 Natural Language Process(NLP)와 같은 분야에서 딥러닝의 활약에 따라 네트워크 트래픽 분류에서도 딥러닝이 등장하게 되었다. 딥러닝의 내부적인 발전 과정에 따라 수 많은 피처들 가운데 특정 피처에 대한 오버피팅을 방지하고, 학습 일반화를 하기 위해 Dropout, Batch nomalization 방법이 일반화되면서 보다 높은 성능을 나타내어 지속적으로 성장하였다. 초기에는 레이어를 쌓아서 인공신경망을 구성하는 Automatic Encoder(AE)[6]가 사용되기도 했으나, 딥러닝의 발전에 맞춰서 컴퓨터 비전분야에서의 대표적인 딥러닝기법인 Convolutional Neural Network(CNN)[7], 자연어처리 분야에서의 Recurrent Neural Network(RNN)[8]를 활용하는 연구들이 연이어 등장하면서 암호화된 트래픽 분류 정확도가 90%를 상회하는 높은 성능을 보이게 되었다. 이러한 가운데 앞서 시계열적 연속성이 있는 자연어처리에서 RNN보다 발전된 Transformer모델[9]이 등장하면서 딥러닝의 도약이 시작되었다. Transformer 모델은 자연어처리에서 뿐만이 아니라 다시금 컴퓨터 비전 분야에서까지 그 유용성이 확인되었다. 그럼에 따라 시계열적 연속성이 있는 패킷들로 구성된 네트워크 트래픽에도 적합할 것이라는 기대와 함께 네트워크 트래픽분류 분야에서도 이를 활용한 연구[10][11]들이 시작되었다.

본 논문에서는 Transformer를 기반으로 발전한 Bidirectional Encoder Representations from Transformers (BERT)모델[12]을 경량화한 DistilBERT모델[13]과 경량화한 데이터를 활용해 트래픽 분류에 적용했으며, 논문에서 제시하는 분류 방법이 가지는 주요한 연구성과는 다음의 3가지로 볼 수 있다.

1) 헤더정보와 암호화된 페이로드 일부 정보로 데이터를 경량화하고 모델에 반영해, 별도의 사전학습 과정 없이 짧은 파인튜닝시간 만으로 높은 분류성능을 나타내었다.
   트래픽 데이터의 첫 번째 패킷에서 100bytes (패킷 단위)와 양방향 패킷으로 구성된 플로우에서 앞 5개의 패킷 각각 100bytes만을 연결(플로우 단위)해 입력으로 사용했다. 6개 클래스의 데이터에서 패킷 단위 분류정확도는 0.9705, 플로우단위는 0.9703을 보였다. 20 에포크의 학습 시간이 1시간 40분이 채 걸리지 않았다.
2) 경량화한 트래픽 데이터에 맞춰서 분류에 사용하는 모델도 자연어처리분야의 기존 BERT 모델이 경량화된 DistilBERT모델을 활용해, 효율적인 트래픽 분류가 가능하도록 했다. 분

류하는데 소모되는 시간은 0.0099 second/packet임에 따라 패킷 단위 분류를 적용해 근실시간에 가까운 사용이 가능하며 하드웨어의 성능에 따라서 보다 향상될 수 있다.

3) Ground-truth 특성이 있는 공개 데이터 셋을 사용함으로써 실제 환경에서의 클래스 불균형 특성을 가진 데이터를 그대로 반영했다.

여기에 Class Weight를 적용함으로써 학습하는 과정이 안정적이고 빠르게 진행되도록 함으로써, 활용성 측면에서 클래스 불균형성이 있는 새로운 데이터에 비교적 빠르게 모델을 적용할 수 있도록 했다.

본 논문의 구성은 다음과 같다. 이어지는 제2장은 관련 연구와 연구 배경으로 세부 구분하여 구성되어 있다. 제3장은 데이터 전처리, 실험 준비 및 모델에 대한 설명으로 구성되어 있다. 제4장에는 모델을 활용한 트래픽분류 결과와 이에 대한 분석이 제시되어 있으며, 마지막으로 제5장에는 결론 및 향후 연구 방향이 기술되어 있다.

Ⅱ. 관련연구 및 연구배경

본 연구에서는 데이터와 딥러닝 모델 경량화에 중점을 두고 있는 만큼, 암호화된 트래픽분류와 관련된 딥러닝 기법 연구들에 대해 집중적으로 알아보았다. 이어서 데이터와 모델 각각에 대해 어떠한 배경으로 경량화 측면에서 연구가 시작되었는지 동기부여 관점 순서로 중점을 세부 구분해 살펴보았다.

2.1. 관련연구

2.1.1 CNN, RNN 모델

CNN과 RNN을 결합하여 조합된 딥러닝 모델이 네트워크 트래픽 분류에 적용되는 연구도 제시되었다. [7]에서 데이터로는 20개의 연속된 패킷으로 구성된 플로우를 기반으로 하는데, 각각의 패킷은 6개의 피처(source port, destination port, 페이로드 크기, TCP window size, 패킷 도착시간, 패킷 방향)로 구성된다.

모델 측면에서는, 시계열 특성을 보존한 데이터가 CNN모델을 거쳐서 특징 벡터가 생성하게 되는데, 이 특징 벡터가 Long Short-Term Memory (LSTM) 계층의 내부 히든레이어의 크기와 동일한 행렬벡터임에 따라 입·출력이 동일하게 적용되면서 Fully Connected(FC) layer로 연결되는 구조를 형성하였다. 이러한 구조를 통해 20개의 연속된 패킷을 분류에 적용해, 정확도 0.9632를 달성했다.

일반적으로 CNN 모델은 여러 개의 convolution layer, pooling layer 및 FC layer로 구성되어 있으며, 이전 레이어의 출력이 다음 레이어의 입력으로 사용된다. 여기서 출력 생성에 사용되는 커널로 동일한 크기를 적용함에 따라 학습가능한 파라미터의 수가 줄어들고, 이러한 특성에 따라 패턴을 캡처하는 커널은 위치에 관계 없이 패턴을 찾아낼 수 있다는 중요 점을 드러낸다.
이러한 점은 특정 패턴이 존재하고, 그 특정 패턴의 이동 불변성이 네트워크 트래픽에서도 마찬가지로 적용 가능한 것[14]으로 연구되었다.

2.1.2 Transformer 모델

네트워크 트래픽은 시계열적 특성을 가지고 있음에 따라 이를 시퀀셜한 구조를 가지는 RNN 모델을 적용하면 우수한 성능을 보였다.
그러나 주목받았던 RNN은 자체의 구조적 특징으로 인해 학습 간 획득한 attention weight이 학습진행에 따라 그 값이 손실되는 구조를 가진다는 단점이 제기됐다. 이에 positional encoding, self-attention 및 multi-head attention을 핵심으로 하는 Transformer 모델[9]이 대안으로 제시되었다.

가장 중요한 점은 positional encoding을 사용하여 입력된 데이터의 순서정보가 보존되는데, 같은 입력 값이 다른 위치에 입력되면 그 순서를 반영해 다른 임베딩 벡터값을 가지게 한다는 것이다.

self-attention은 자기 자신에게 주의 집중한다는 의미로서, positional encoding을 거쳐서 입력된 입력 데이터들 간에 유사도를 구함으로써 연관성을 찾아내는 것이다.

multi-head attention은 한 번만 어텐션 헤드를 사용 것 보다 여러 번 병렬로 사용되는 것이 효과적이라는 사실하에, 8개의 병렬 어텐션 행렬로부터 다양한 시각에서의 정보를 얻는다. 그리고 모든 어텐션 헤드를 연결하여 종합된 정보를 얻어내는 구조로 되어 있다. 이러한 Transformer 모델의 self-attention 구조로 패킷 하나만을 분류에 적용한 연구[8]에서 정확도 0.9480의 성능을 보였다.

2.1.3 BERT 모델

Transformer 구조를 기반으로 특화된 BERT는

그 이름에서도 알 수 있듯이 Transformer의 인코더를 활용해 입력 데이터를 양방향 관점에서 인식하는 것이다. BERT에는 입력으로 3개의 벡터들이 동시 들어가게 된다.

첫 번째는 각 문장이 토큰 단위로 구분되는 embedding 벡터이고, 두 번째는 Position에 대한 벡터이며, 마지막은 토큰이 어느 문장에 속하는지에 대한 segmentation 벡터이다.

이렇게 입력된 3개의 벡터를 두 가지 방법으로 사전학습 하는 메커니즘을 가지고 있다.
하나는 입력 데이터를 15%의 확률로 단어를 선택하여 [MASK]토큰을 생성한 뒤 원래의 단어를 예측하도록 학습하는 것이고, 또 다른 하나는 두 개의 문장을 입력으로 주어 각 문장의 연관성을 학습하는 것이다.

이러한 BERT모델을 기반으로 암호화된 페이로드 전체를 사전학습 과정과 파인튜닝 과정을 통해 패킷 5개와 패킷 1개를 입력 데이터로 각각 정확도 0.9729, 0.9890을 달성[15]했다. 오랜 사전학습 과정과 패킷을 통째로 입력데이터로 사용하는 만큼, 연산에 필요한 자원과 시간이 막대했을 문제점이 예상되었다.

2.2. 연구배경

2.2.1 데이터 경량화

연대기적 관점에서 우선적으로 눈여겨 볼만한 연구 중에 하나는 직관적으로 처음 몇 개의 패킷이 애플리케이션의 연결단계에서의 중요한 정보를 담고 있기 때문에 트래픽분류의 중요한 점으로 짚어낸 연구[4]이다. 각 TCP 플로우의 처음 5개 패킷을 사용하여 여러 애플리케이션에 대해 전체 플로우의 80% 이상의 정확도를 보여주었다.

다만 예외적으로 POP3 애플리케이션의 경우는 클러스터를 구성하지 않는 점 때문에 정확도가 떨어지지만, 이 연구의 결과에 따라 트래픽의 플로우를 조기에 감지하는데 좋은 교훈을 배우게 된 점은 이전부터도, 지금도[15] 주목받고 있다. 따라서 우리는 여기에 기반하여 플로우의 처음 5개 패킷으로 이루어진 플로우를 입력 데이터로 활용했다.

또한 트래픽 분류모델에서 새롭지만 반드시 필요한 실시간 트래픽분류를 하기 위한 시도가 등장했다. 여러 개의 패킷이 연결된 플로우 수준의 데이터를 입력으로 받지 않고, 단위 패킷 수준만을 데이터로 입력받는 모델을 제안한 연구가 제시되었다.

[11]에서는 트래픽 분류를 위해 애플리케이션 계층에서 헤더 정보 외에도 페이로드를 일부만 사용해도 페이로드에 잠재적으로 민감한 사용자 활동 정보가 포함되어 있기 때문에 정보보호의 문제를 제기하면서 페이로드 첫 40, 50, 60bytes 중에 분류 성능 및 보안적 관점에서 최적값을 찾고자 하였다. 실험을 통해 40bytes에 대비해서 50bytes는 성능향상이 크게 있지만, 60bytes는 성능적 향상이 없음을 제시했다. 성능과 보안적 측면에서 헤더 정보 외에 추가적으로 페이로드 50bytes만을 추가적으로 입력 데이터의 최적값으로 선택하기도 했다.

2.2.2 경량화된 모델

BERT는 뛰어난 성능과 간단한 파인튜닝 기법에도 불구하고 거대한 모델크기(파라미터 개수), 느린 추론 속도, 복잡하고 비용이 많이 드는 사전학습 과정으로 인해 사용이 제한되어 모델을 경량화하고 추론 속도를 높이고자 하는 많은 연구가 있었다.

많은 head들을 제거하더라도 성능에 영향이 없다는 것을 밝혀낸 연구[16]도 있었다.
자연어처리 분야에서 Transformer를 기반으로 하는 모델에서 사용되는 multi-head attention은 각각의 head는 입력의 각기 다른 부분에 집중하도록 한다. 결과적으로 단순히 가중합을 사용하는 것보다 정교한 함수를 사용해 정보를 가공할 수 있다는 것에 대한 반론을 제시한 것이다.

성능이 좋고 크기가 큰 교사 모델의 결과를 작은 학생모델에 가르치는 접근방법인 Knowledge Distillation(KD)[17]이 등장하면서 BERT 경량화의 흐름이 바뀌었다. 각 로짓에 대해 temperature를 조절함으로써 일반적인 소프트맥스에 비해 정답이 아닌 라벨에도 작은 확률값을 부여함으로써 테스트 데이터에서 일반화 성능을 높이는 방법이었다.

이러한 KD연구에 기반하여 BERT의 크기를 40%까지 줄이면서도, 성능은 97%를 유지하고, 추론속도는 60% 정도 향상된 실효성을 가지는 것이 DistilBERT 모델 연구에 제시되어 있다.
따라서 본 논문에서는 트래픽 데이터를 경량화하고, 이렇게 경량화된 데이터를 NLP분야에서 우수한 성능을 가지는 BERT 모델이 경량화된 DistilBERT a 모델에 적용하였다. NLP분야의 모델을 네트워크 트래픽 분류에 맞춰 활용하고자 한 것이다.

Ⅲ. 데이터 및 모델 구성

3.1. 데이터셋 선정

인터넷에 공개되어 다른 논문들에서도 공통적으로 활용되고 있는 데이터셋인 VPN-nonVPN dataset (ISCXVPN2016)[17]을 선택했다.

그 이유 중 하나는, 실제 사용되는 데이터를 수집해 Ground-truth 특성을 가지고 있다는 점이다. 또 다른 이유는 이전에 이루어진 많은 논문들이 제시한 실험결과와 비교가 가능하여 본 연구가 실질적인 의미에서 가치 있을 것으로 판단했기 때문이다.

공개데이터 분포 비율을 유지한 채 샘플데이터로 6개 클래스의 데이터를 구성한 것인데, 학습과 분류에 소요되는 자원과 시간을 고려한 선택이었다. 아래의 표 1은 트래픽 분류범위 안에 포함된 어플리케이션의 종류를 나타낸 것이다.

표 1. 트래픽 분류범위 내 애플리케이션의 구성
Table 1. Composition of applications within traffic catagories

Traffic Categories	Applications
Email	SMTP/S, POP3/SSL and IMAP/SSL
Chat	AIM, Facebook, Gmail, Hangouts, ICQ and Skype
Streaming	Netflix, Spotify, Vimeo and Youtube
File Transfer	FTPS, SCP, SFTP and Skype
VoIP	Facebook, Skype, Hangouts(voice and video calls) and VoipBuster
P2P	Bittorrent

다음 그림 1은 6개 클래스의 데이터 27,811개로 구성되어 있으며, 그에 대한 수량적 비율 분포도를 파이차트로 나타낸 것이다. 각 클래스가 균형적이지 않고 사용자의 사용성에 따라 각 클래스별 데이터가 다른 것이 특징적이다.

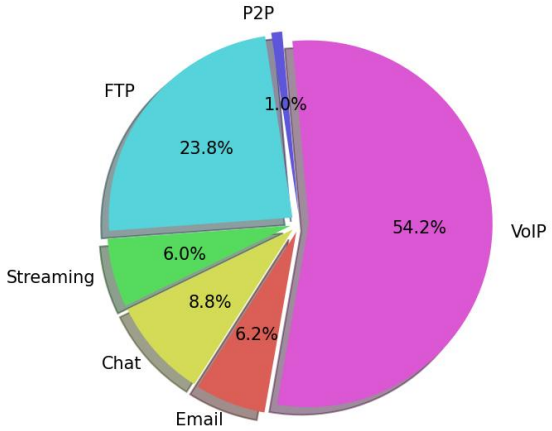

그림 1. 6개 클래스로 구성된 27,811개의 데이터 분포비율
Fig. 1. Distribution ratio of 27,811 data composed of 6 classes

3.2. 데이터 전처리 및 리프젠테이션

[18], [19]에서 모델에 입력되기 전에 로컬 최소값을 상위 수준의 추상화인 내부 분산표현을 생성함으로써 최적화에 도움이 된다는 연구결과를 제시하였다. 따라서 로컬 최소값을 의미하는 데이터에 대한 전처리 및 리프리젠테이션은 딥러닝에서 중요한 역할을 한다.

그래서 우리는 [20]에서 사용하는 방법을 준용하여 ethernet header와 IP header에 대한 처리, DNS 또는 페이로드가 없는 패킷에 대한 처리, 패딩 및 정규화를 적용했다. 전처리 알고리즘은 아래 그림 2의 알고리즘 1과 같다.

Algorithm 1. Proprocessing

```
Input : PATH = continuous bi-directional packets, MAX_PACKET = 100, MAX_ROW = 5
Output : packets_row = preprocessed data from pcap file
1 : packets_row = [ ]
2 : for i, packet in enumerate(read_pcap(PATH))
3 :   if i >= 0 and i < MAX_ROW
4 :     packet = discard_packet(packet)        # no payload TCP or DNS
5 :     packet = remove_ethernet_header(packet)
6 :     if udp == categorize_packet(packet)        # distinguish protocol
7 :         packet = pad_udp(packet)        # with zeros to length of 20bytes
8 :     packet = mask_ip(packet)        # src, dst : 0.0.0.0
9 :     packet = normalization(packet)
10 :                # min-max scale with values between 0 and 1 for each element
11 :                # fix the packet into MAX_PACKET bytes using zero_padding or cutting
12 : return packets_row        # for saving the proprocessed data
```

그림 2. 전처리 알고리즘
Fig. 2. Preprocessing algorithm

모델에 입력하기 위한 단위로 패킷 1개는 앞 100bytes로 구성되어 있으며, 양방향 플로우(세션)인 패킷 5개의 100bytes가 연결된 구성이다.

데이터셋을 train과 test로 분할시 비율은 8:2로 처리했다.

3.3. 실험전 셋팅

앞에서 제시된 데이터의 클래스 불균형은 모델의 학습과정에서 오버피팅을 유발할 수 있기 때문에 학습 전에 사전 처리가 필요하다.

[21]에는 클래스 불균형 문제가 언급되어 있는데, 주 표본의 분류에만 집중하면서 소수 표본을 무시하거나 오분류하는 실수가 이루어질 수 있어서 균형을 맞추는 3가지 범주의 방법이 제시됐다. 우리는 그중에 클래스 가중치를 사용하여 학습 과정에서 서로 다른 클래스 분포의 균형을 맞추어 학습하는 방법을 선택했다.

계산식은 아래 (1)과 같으며, 이를 적용해 얻은 상대적인 클래스 웨이트 값은 그림 3과 같다.

$$W = 1 - \frac{Q_i}{\sum_i Q_i} \quad (1)$$

W는 클래스 웨이트, Q_i는 i번째 클래스의 데이터 수량을 의미한다.

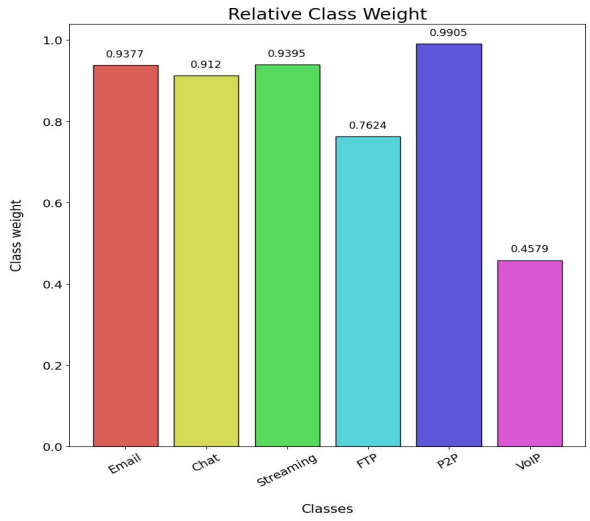

그림 3 상대적 클래스 웨이트

Fig. 3. Relative Class Weight

모델의 학습과정에 클래스 웨이트가 미치는 영향을 비교한 실험은 '4.3 추가적인 실험과 분석' 부분에 제시되어 있다.

3.4. 모델 구성

전처리과정을 통해 리프리젠테이션된 패킷이 토크나이저를 거쳐 DistilBERT를 통해 학습 및 분류되는 과정을 아래 그림 4와 같이 단계적으로 나타내었다.

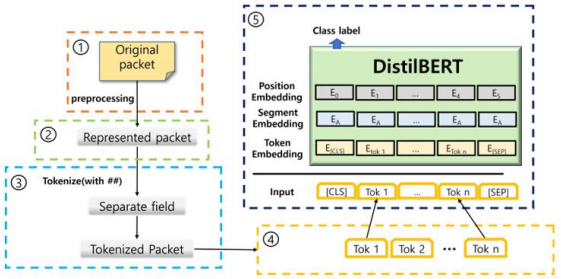

그림 4. 전체적인 모델 구성

Fig. 4. Overall model configuration

순차적인 진행 과정은 다음과 같다.

1) 전처리 과정을 통해 로컬 최소값을 상위 수준의 추상화인 내부 분산표현으로 생성함.
2) 전처리시 오버피팅을 없애고 모델의 학습효과를 높이도록 리프리젠테이션도 동시수행 함.
3) bytes 단위로 구분된 각 필드의 값을 토크나이징하여 vocabulary화 함.
4) 입력의 맨 앞, 뒤에 스페셜 토큰을 추가하여 최종 토큰들로 최종 변환함.
5) 최종 입력 데이터를 사용해 DistilBERT 모델이 학습하고 추론함.

표 2. 모델 내부의 순차적인 데이터 구조

Table 2. Sequential data structure inside the model

Step	Example Data	Size of Data
①	A = [d_1, … ,d_{100}]	100 bytes
②	A′ = [d'_1, … ,d'_{100}]	
③	A″ = [d''_1, … ,d''_{100}]	
④		
⑤	A‴ = [CLS, d''_1, … , d''_{98}, SEP]	100 bytes = 98 bytes (represented data) + 2 bytes (special token)

앞의 표2는 첫 패킷의 앞 100bytes의 최초의 입력 데이터가 전처리 과정에서부터 모델에 입력되기까지의 데이터가 가지는 구조와 크기를 예시적으로 나타낸 것이다.

모델의 학습이 효과적으로 이루어지도록 선택한 옵티마이저는 RAdam이다. RAdam[21]은 Adam[22]이 가지고 있는 bad local optima problem을 해결하고자 Adam의 adaptive learning rate term의 분산을 바로잡아, 적은 학습 단계에서도 학습 안정성을 높인 옵티마이저이다. RAdam을 사용함으로써 학습 단계(에포크)에서 빠르고 안정성 있게 학습하는 성능은 '4장 실험 결과 및 분석' 부분에 제시되었다.

Ⅳ. 실험 결과 및 분석

4.1. 실험 환경

모델은 CUDA 11.3 기반의 TensorFlow 2.9.2, Python 3.7.14 및 Pytorch 1.12.1를 기반으로 구현되었다. 하드웨어는 4Core Intel(R) Xeon(R) CPU @ 2.30GHz의 CPU와 NVIDIA Tesla P-100(16GB 메모리)의 GPU가 장착된 서버로 구성되었다.

4.2. 실험결과 및 분석

모델을 평가하기 위한 요소로 사용된 지표들은 아래 (2), (3), (4) 및 (5)입니다.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \quad (2)$$

$$\mathrm{Precision} = \frac{TP}{TP + FP} \quad (3)$$

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

$$F1 - score = 2 \times \frac{\mathrm{Precision} \times Recall}{\mathrm{Precision} + Recall} \quad (5)$$

위에서부터 순서대로 식 (2)에 나오는 용어의 의미이다. True Positive(TP)는 실제 True인 정답을 True라고 예측(정답)하는 것이다. False Positive(FP)는 실제 False인 정답을 True라고 예측(오답)하는 것이다. False Negative(FN)는 실제 True인 정답을 False라고 예측(오답)하는 것이다. 마지막으로 True Negative(TN)는 실제 False인 정답을 False라고 예측(정답)하는 경우이다.

다음 그림 5, 6은 우리 모델의 학습을 통한 예측 성능을 나타낸 컨퓨전 메트릭스와 각 평가요소이다.

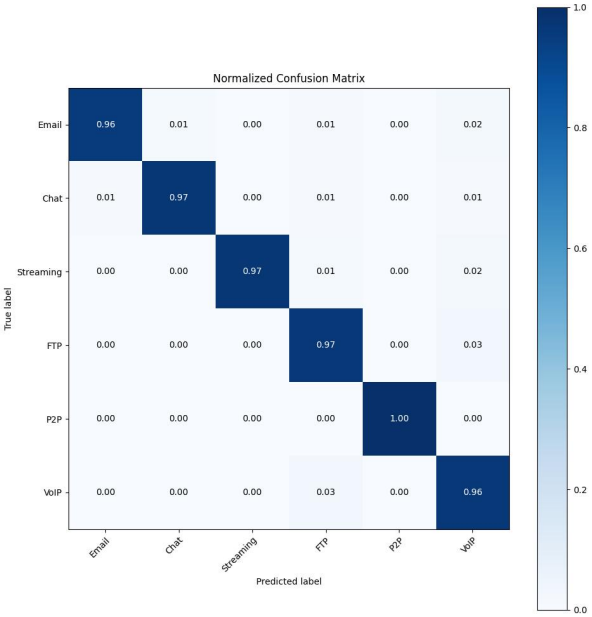

그림 5. 컨퓨전 메트릭스
Fig. 5. Confusion Matrix

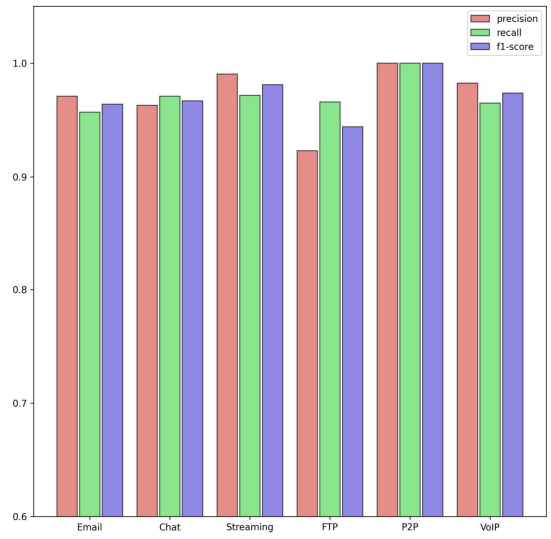

그림 6. 정밀도, 재현율, F1-score
Fig. 6. Precision, Recall, F1-score

그림 5에서 보면, 6개 클래스의 트래픽을 분류한 결과 모두 0.96 ~ 1.0 값을 나타내어 Accuracy가 고르게 높은 모습이 보인다. 그림 6에서 보면, FTP 트래픽을 제외하고는 5개 클래스 모두 Precision, Recall, F-1-score 모두 고르게 0.95 이상을 상회하는 모습을 볼 수 있다.

아래의 표3는 동일한 데이터셋인 VPN-nonVPN dataset (ISCXVPN2016)을 사용하는 다른 모델들과 정확도와 F1-score 값을 통해 성능을 비교한 것이다. [8]에서는 1500bytes 길이의 패킷 10개로 구성된 데이터를 LSTM과 Attention Mechanism(AM)이 조합된 모델에 적용해 얻은 성능이다. 더 긴 패킷 길이와 더 많은 패킷 수를 사용했음에도 우리의 성능에 못 미친다.
[11]에서는 90bytes(헤더 40bytes + 페이로드 50bytes)길이로 구성된 데이터를 self-attention이 적용된 모델에 적용해 얻은 성능이다. 조금 짧은 패킷 길이를 적용했지만 우리의 성능에 크게 못 미치는 것을 확인할 수 있다.

표 3. 동일한 데이터셋을 활용한 다른 연구들과의 성능비교 (정확도, F1-score)
Table 3. Performance comparison with other studies using the same dataset

	Unit	# of classes	Accuracy / F1-score
[8]	session	12	0.9120 / -
		6	0.9480 / -
[11]	packet	6	0.9033 / 0.8560
Proposed	session	6	**0.9703 / 0.9706**
	packet		**0.9707 / 0.9731**

표 4에서는 모델의 총 파라미터 개수 측면에서 우리가 선택한 DistilBERT가 일반적인 BERT모델이 가지는 파라미터보다 훨씬 적은 수의 파라미터를 사용하는 것을 확인할 수 있다. 경량화된 데이터로 DistilBERT를 활용해 파인튜닝하는 학습과정 후, 테스트 시간은 0.0099 second/packet임에 따라 패킷 단위 분류에 적은 시간이 소요됨을 확인했다.

표 4. 모델의 총 파라미터 갯수
Table 4. Total number of parameters in model

	BERT	DistilBERT
# of parameter	177,853,440	**66,965,007**

4.3. 추가적인 실험과 분석

그리고 멀티클래스 데이터가 클래스 불균형을 이룰 때 모델의 성능을 비교하는 지표는 Accuracy가 아닌 F1-score임에 따라, 추가적으로 표 5의 성능지표를 확인했는데, 모델의 뛰어난 학습성능으로 인해 결과적 수치로는 큰 차이가 없었다.
그러나 로스 함수에 클래스 웨이트를 적용하는 여부에 따른 학습 곡선을 아래의 그림 7, 8에서 비교해보면, 클래스 웨이트가 적용된 모델의 학습 곡선이 비교적 안정적인 모습임을 확인할 수 있다.

표 5. 로스함수에 클래스 웨이트 적용여부에 따른 성능비교
Table 5. Performance comparison according to whether or not class weight is applied to the loss function

	Class Weight	Non Class Weight
Accuracy / F1-score	0.9705 / 0.9746	0.9703 / 0.9742

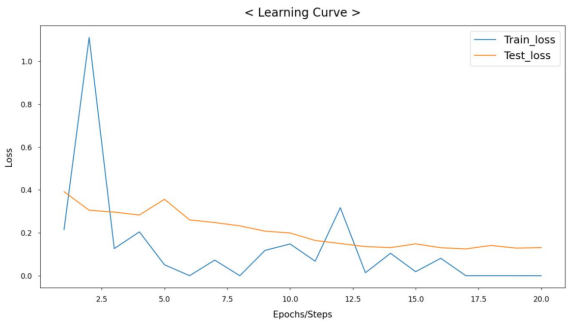

그림 7. 클래스 웨이트를 적용한 모델의 학습 곡선
Fig. 7. Learning curve of a model with class weights

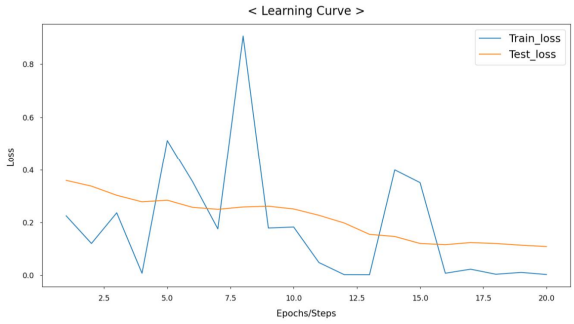

그림 8. 클래스 웨이트를 미적용한 모델의 학습 곡선
Fig. 8. Learning curve of a model without class weights

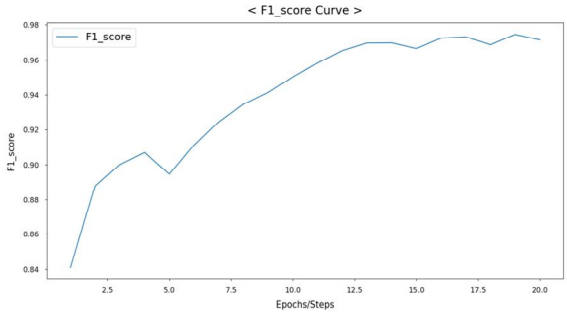

그림 9. 클래스 웨이트를 적용한 모델의 F1-score 곡선
Fig. 9. F1-score curve of the model with class weights applied

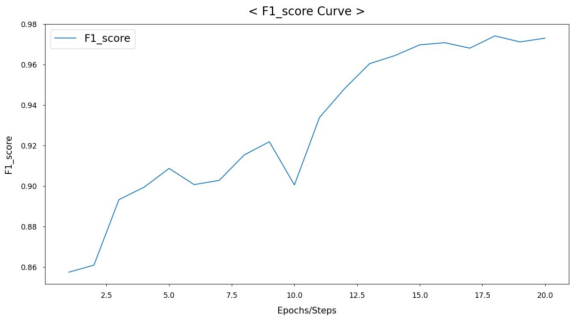

그림 10. 클래스 웨이트를 미적용한 모델의 F1-score 곡선
Fig. 10. F1-score curve of the model without class weights

동일한 관점에서 로스 함수에 클래스 웨이트를 적용하는 여부에 따라, F1-score 값의 변화 곡선을 위의 그림 9, 10을 통해 비교해 보았다. 클래스 웨이트가 적용된 모델의 F1-score 값이 보다 빠르고 안정적이게 증가함을 확인할 수 있다.

또한 실험 간에 확인된 사실은 27,811개의 데이터 중 80%로 구성된 22,248개의 학습데이터로 1에포크의 training에 소요된 시간은 4분 45초였고, 사전학습이 불필요했다는 점을 감안해 보면 굉장히 짧은 학습 시간임을 알 수 있다.

V. 결론 및 향후 연구방향

5.1. 결론

트래픽분류는 컴퓨터 네트워크 영역에서 서비스 관리 및 보안 등의 분야에서 그 역할이 점점 더 중요해지고 있다. 초기에는 포트넘버, DPI, 통계정보 등을 활용해 트래픽 분류가 가능했다. 그러나 정보보호 측면에서 트래픽의 페이로드가 암호화되면서 앞서의 기법들로는 좋은 성능을 달성할 수 없었다. 그에 따라 머신러닝기법을 추가 활용하면서, 여러 가지 조합에 따라 선택된 변수(통계정보, 시그니처 등)들과 함께 트래픽분류는 한걸음 나아갔다. 이후 AE, CNN, RNN 등의 다양한 딥러닝 모델들을 활용한 트래픽 분류도 계속 발전되었다.

이후 RNN의 단점을 개선한 Transformer 모델이 self-attention mechanism이 제시되었고, 매우 우수한 성능을 보여줌에 따라 딥러닝은 또 한 걸음 나아갔다. 따라서 NLP분야에서 우수한 성능을 보이는 Transformer를 기반으로 다양한 분야에서 딥러닝이 확장되고 있다.

하지만 딥러닝에 계속 발전하면서 장점이자 단점으로, 변수 선택은 그 역할이 줄어들고 수 많은 변수들로 구성된 데이터를 활용해도 트래픽 분류가 문제없어짐에 따라, 딥러닝 모델과 데이터가 점차 커졌다. 이에 따라 큰 모델에는 자원과 시간이 많이 소모됨에 따라 부담스럽고 성가시게 되었다.

이렇게 부담스럽고 성가신 모델과 데이터를 경량화하여 효율적인 네트워크 트래픽 분류가 필요했다. 본 연구에서는 KD 연구결과에 따라 제시되었던 모델 경량화 기법이 적용된 DistilBERT 모델에 경량화한 데이터를 적용하는 것이 중요하다고 판단했다. 그리고 모델의 선택에 있어서 동시적으로 고려된 점으로, 트래픽 데이터가 시계열적 연속성을 가짐에 따라 NLP모델을 트래픽분류에 적용하는 것이 가치 있을 것이라는 점이다.

이에 따른 실험결과로 플로우의 첫 번째 패킷 1개의 앞 100bytes 입력과 그 패킷 5개를 연결한 입력으로 정확도 / F1-score가 각각 0.9707 / 0.9731과 0.9703 / 0.9706를 보여, 현재까지 제시된 다른 연구들에 대비해 매우 우수한 성능임을 확인했다.

5.2. 향후 연구방향

이후에 암호화된 네트워크 트래픽 분류분야에 딥러닝을 적용하는 측면에서, 우리의 연구 방향은 크게 다음 3가지 정도로 판단하고 있다.

첫째, 패킷과 플로우를 NLP관점에서 처리하는 방법과 이미지관점에서 처리하는 방법을 적용해 보고 장·단점을 비교해 보는 것이다. 두 번째, 전처리과정에서 암호화된 페이로드가 가지는 함축된 정보를 전달할 수 있도록 요약적 페이로드를 적용해 보는 것이다. 마지막으로 입력데이터인 패킷과 플로우를 모델에서 보다 효율적으로 인식할 수 있도록 모델의 메커니즘을 개선해 보는 것이다.

References

[1] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, “Characterization of Encrypted and VPN Traffic using Time-related Features:,” in Proceedings of the 2nd International Conference on Information Systems Security and Privacy, Rome, Italy, 2016, pp. 407 –414. doi: 10.5220/0005740704070414.

[2] A. W. Moore and K. Papagiannaki, “Toward the Accurate Identification of Network Applications,” in Passive and Active Network Measurement, vol. 3431, C. Dovrolis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 41–54. doi: 10.1007/978-3-540-31966-5_4.

[3] T. T. T. Nguyen and G. Armitage, “A survey of techniques for internet traffic classification using machine learning,” IEEE Communications Surveys Tutorials, vol. 10, no. 4, pp. 56–76, 2008, doi: 10.1109/SURV.2008.080406.

[4] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, “Traffic classification on the fly,” SIGCOMM Comput. Commun. Rev., vol. 36, no. 2, pp. 23–26, Apr. 2006, doi: 10.1145/1129582.1129589.

[5] J. Erman, A. Mahanti, M. Arlitt, and C. Williamson, “Identifying and discriminating between web and peer-to-peer traffic in the network core,” in Proceedings of the 16th international conference on World Wide Web - WWW ’07, Banff, Alberta, Canada, 2007, p. 883. doi: 10.1145/1242572.1242692.

[6] J. Höchst, L. Baumgärtner, M. Hollick, and B. Freisleben, “Unsupervised Traffic Flow Classification Using a Neural Autoencoder,” in 2017 IEEE 42nd Conference on Local Computer Networks (LCN), Oct. 2017, pp. 523–526. doi: 10.1109/LCN.2017.57.

[7] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, “Network Traffic Classifier With Convolutional and Recurrent Neural Networks for Internet of Things,” IEEE Access, vol. 5, pp. 18042–18050, 2017, doi: 10.1109/ACCESS.2017.2747560.

[8] H. Yao, C. Liu, P. Zhang, S. Wu, C. Jiang, and S. Yu, “Identification of Encrypted Traffic Through Attention Mechanism Based Long Short Term Memory,” IEEE Transactions on Big Data, vol. 8, no. 1, pp. 241–252, Feb. 2022, doi: 10.1109/TBDATA.2019.2940675.

[9] A. Vaswani et al., “Attention is all you need,” in Proceedings of the 31st International Conference on Neural Information Processing Systems, Red Hook, NY, USA, Dec. 2017, pp. 6000–6010.

[10] HaoLi, “Traffic classification algorithm using CNN and multi-head attention mechanism for representation learning,” J. Phys.: Conf. Ser., vol. 2258, no. 1, p. 012001, Apr. 2022, doi: 10.1088/1742-6596/2258/1/012001.

[11] G. Xie, Q. Li, and Y. Jiang, “Self-attentive deep learning method for online traffic classification and its interpretability,” Computer Networks, vol. 196, p. 108267, Sep. 2021, doi: 10.1016/j.comnet.2021.108267.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” arXiv, May 24, 2019. doi: 10.48550/arXiv.1810.04805.

[13] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.” arXiv, Feb. 29, 2020. Accessed: Sep. 15, 2022. [Online]. Available: http://arxiv.org/abs/1910.01108

[14] S. Rezaei and X. Liu, “Multitask Learning for Network Traffic Classification,” in 2020 29th International Conference on Computer Communications and Networks (ICCCN), Aug. 2020, pp. 1–9. doi: 10.1109/ICCCN49398.2020.9209652.

[15] X. Lin, G. Xiong, G. Gou, Z. Li, J. Shi, and J. Yu, “ET-BERT: A Contextualized Datagram Representation with Pre-training Transformers for Encrypted Traffic Classification,” Feb. 19, 2022. Accessed: Apr. 02, 2022. [Online]. Available: http://arxiv.org/abs/2202.06335

[16] P. Michel, O. Levy, and G. Neubig, “Are

Sixteen Heads Really Better than One?" arXiv, Nov. 04, 2019. Accessed: Dec. 02, 2022. [Online].Available:http://arxiv.org/abs/1905.10650

[17] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network." arXiv, Mar. 09, 2015. Accessed: Sep. 16, 2022. [Online].Available:http://arxiv.org/abs/1503.02531

[18] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy Layer-Wise Training of Deep Networks," in Advances in Neural Information Processing Systems, 2006, vol. 19. Accessed: Dec. 02, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2006/hash/5da713a690c067105aeb2fae32403405-Abstract.html

[19] G. Zhong, L.-N. Wang, and J. Dong, "An Overview on Data Representation Learning: From Traditional Feature Learning to Recent Deep Learning." arXiv, Nov. 24, 2016. Accessed: Dec. 02, 2022. [Online]. Available: http://arxiv.org/abs/1611.08331

[20] M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, and M. Saberian, "Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning," arXiv:1709.02656 [cs], Jul. 2018, Accessed: Mar. 16, 2022. [Online]. Available: http://arxiv.org/abs/1709.02656

[21] L. Liu et al., "On the Variance of the Adaptive Learning Rate and Beyond." arXiv, Oct. 25, 2021. doi: 10.48550/arXiv.1908.03265.

[22] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization." arXiv, Jan. 29, 2017. doi: 10.48550/arXiv.1412.6980.

신 창 의 (Chang-Yui Shin)

2003년 : 육군사관학교 운영분석학과 학사

2007년 : 고려대학교 전자컴퓨터 공학과 석사

2022년~현재 : 고려대학교 컴퓨터정보학과 박사과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 분석

박 지 태 (Jee-Tae Park)

2017년 : 고려대학교 컴퓨터정보학과 학사

2017년~현재 : 고려대학교 컴퓨터정보학과 석박사통합과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 분석

백 의 준 (Ui-Jun Baek)

2018년 : 고려대학교 컴퓨터정보학과 학사

2018년~현재 : 고려대학교 컴퓨터정보학과 석박사통합과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

최 정 우 (Jung-woo Choi)

2018년 : 고려대학교 컴퓨터정보학과 학사

2022년~현재 : 고려대학교 컴퓨터정보학과 석사과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

김 명 섭 (Myung-Sup Kim)

1998년 : 포항공과대학교 전자계산 학과 학사

2000년 : 포항공과대학교 전자계산 학과 석사

2004년 : 포항공과대학교 전자계산 학과 박사

2006년 : Dept. of ECS, Univof Toronto Canada

2006년~현재 : 고려대학교 컴퓨터정보학과 교수

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석, 멀티미디어 네트워크