Recurrent Neural Network을 이용한 플로우 기반 네트워크 트래픽 분류

임현교*, 김주봉**, 허주성**, 권도형*, 한연희**†
*한국기술교육대학교 창의융합협동과정
**한국기술교육대학교 컴퓨터공학과
e-mail:{glenn89, rlawnqhd, chil1207, dohk, yhhan}@koreatech.ac.kr

Flow based Network Traffic Classification Using Recurrent Neural Network

Hyun-Kyo Lim, Ju-Bong Kim, Joo-Seong Heo, Do-Hyung, Kwon, Youn-Hee Han
Korea University of Technology and Education, Korea

요 약

최근 다양한 네트워크 서비스와 응용들이 생겨나면서, 네트워크상에 다양한 네트워크 트래픽이 발생하고 있다. 이로 인하여, 네트워크에 불필요한 네트워크 트래픽도 많이 발생하면서 네트워크 성능에 저하를 발생 시키고 있다. 따라서, 네트워크 트래픽 분류를 통하여 빠르게 제공되어야 하는 네트워크 서비스를 빠르게 전송 할 수 있도록 각 네트워크 트래픽마다의 분류가 필요하다. 본 논문에서는 Deep Learning 기법 중 Recurrent Neural Network를 이용한 플로우 기반의 네트워크 트래픽 분류를 제안한다. Deep Learning은 네트워크 관리자의 개입 없이 네트워크 트래픽 분류를 할 수 있으며, 이를 위하여 네트워크 트래픽을 Recurrent Neural Network 에 적합한 데이터 형태로 변환한다. 변환된 데이터 세트를 이용하여 훈련시킴으로써 네트워크 트래픽을 분류한다. 본 논문에서는 훈련시킨 결과를 토대로 비교 분석 및 평가를 진행한다.

1. 서론

최근 네트워크에서는 다양한 형태의 응용 및 서비스가 운영되고 있다. 또한 네트워크를 사용하는 사용자의 요구와 사용자가 이용하는 스마트폰, 노트북, 데스크탑 등 어디서나 쉽게 네트워크에 접속 할 수 있게 되고, 요구의 질 또한 증가 하면서 더 크고 다양한 규모의 트래픽 데이터가 발생하고 있다. 따라서 사람이 해야 할 데이터 분석과 관리 범위가 상당히 넓어지고 있다 [1]. 이에 따라 경량화와 자동화가 된 망의 구축이 필요하고 [2], 더불어 최근 각광받는 딥 러닝 기법의 활용도가 높아지고 있다.

딥 러닝 관련 기법에는 대표적으로 Multilayer Neural Network Model (MNN) [3], Convolution Neural Network Model (CNN) [3], Recurrent Neural Network Model [3] 등이 있다. MNN은 약 3개에서 7개까지의 Hidden Layer로 구성된 기본적인 딥 러닝 모델이며, CNN은 지역 수용영역과 Convolution Layer, Pooling Layer 세 단계로 구성되어 있는 뉴럴 네트워크로써 이미지 분석에 대표적으로 쓰이고 있는 딥 러닝 기법이다. RNN은 임의의 순차 데이터(Sequence Data)를 처리하기에 적합한 순환 신경망에 넣어 출력 데이터를 뽑아내는데 출력 데이터는 이전의 연산결과에 영향을 받는 것이 특징이다.

본 논문에서는 플로우 기반의 네트워크 트래픽 분류를 위하여 딥 러닝 기법 중 RNN을 이용한다. 플로우는 5-tuple 이동일한 트래픽들을 모아 놓은 것으로 순차적으로 네트워크 패킷들이 들어 있다. 따라서 순차 데이터를 훈련시키기에 적합한 RNN을 플로우 기반의 네트워크 트래픽 분류에 이용하려고 한다. 이를 위하여 네트워크상에서 모은 플로우 데이터를 RNN을 이용하여 학습 시킬 수 있는 형태로 변환시키는 과정이 필요하다. 이를 위하여 Traffic Data Preprocessing 과정을 통하여 Raw 트래픽 데이터를 각 응용별로 나누는 Traffic Split, Split한 네트워크 트래픽을 학습에 용이한 데이터로 만드는 Learning Data Generation 과정을 거쳐 RNN 학습을 위한 데이터 세트로 생성한다. 이후 RNN 학습을 통해 해당 플로우를 분류하게 된다.

본 논문의 2장에서는 네트워크 트래픽 데이터의 전처리 과정을, 그리고 3장에서는 RNN 실험에 사용한 RNN 모델의 구조에 대해서 설명하고 4장에서는 실험 및 평가를 진행한다.

2. Traffic Data Preprocessing

본 장에서는 트래픽 분류를 수행하기 위하여 사용하는

† 교신 저자: 한연희 (한국기술교육대학교)
이 논문은 2016년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. NRF-2016R1D1A3B03933355)

Traffic data는 Broadband Communication Research Group [4]에서 제공 받은 PCAP 파일을 RNN 학습에 적합하게 처리하는 과정을 소개한다. PCAP (Packet Capture) 파일은 Wireshark, TCPdump등과 같은 프로그램을 이용하여 네트워크 패킷을 캡쳐 하여 PCAP 파일의 형태로 저장한 것이다. 본 논문에서 제안하는 RNN을 이용한 플로우 기반의 네트워크 트래픽 분류를 위해 제공받은 PCAP 파일을 RNN 트래픽 분류기에 분류 가능한 데이터 세트들로 미리 가공할 필요가 있다. 이를 위하여, PCAP 파일을 필터링하고 축약하는 데이터 전처리 과정이 필요하다.

2.1. Traffic Split

원본 PCAP 파일의 크기는 약 59GB이고, 그림 1과 같이 769507개의 플로우가 존재한다. PCAP 파일과 함께 제공

Application	#Flows	#Megabytes
Edonkey	176 581	2 823.88
BitTorrent	62 845	2 621.37
FTP	876	3 089.06
DNS	6 600	1.74
NTP	27 786	4.03
RDP	132 907	13 218.47
NETBIOS	9 445	5.17
SSH	26 219	91.80
Browser HTTP	46 669	5 757.32
Browser RTMP	427	5 907.15
Unclassified	771 667	3 026.57

(그림 1) Broadband Communication Research Group PCAP Data Flow

받은 'packet_default.info' 파일에는, 트래픽 데이터에 대한 labeling이 존재한다. Labeling은 해당 트래픽 플로우에 대한 응용타입, Protocol, 응용이름과 같이 세 분류로 나누어져 있다. Labeling 파일로 인하여 본 논문에서 제안하는 RNN 을 이용한 플로우 기반의 트래픽 분류에서의 Ground Truth에 해당하는 정확한 label을 얻을 수 있었다. 표 1은 'packet_default.info'의 각 Application의 플로우에 대한 5-tuple과 응용타입, 응용이름에i 대한 정보를 나타낸다.

<표 1> info 파일 데이터 헤더

파일명	데이터 헤더
packets_default.info	flow_id#start_time#end_time#local_ip#remote_ip#local_port#remote_port#transport_protocol#operating_system#process_name#labels#-#-#
Application_Layer_payload.info	flow_id#start_time#end_time#Payload#-#-#

데이터 전처리과정을 위하여 플로우수를 기준으로 Application 8종을 선별하였다. 상위 8종의 label 명은, Remote Desktop Protocol (RDP), Skype, SSH, Bittorrent, HTTP-Facebook, HTTP-Google, HTTP-Wikipedia, HTTP-Youtube이고, 해당 label이 붙은 플로우들만을 PCAP 파일에서 선택하였다. 선택된 플로우 내부 패킷의 Application Layer의 페이로드만을 필터링하여 추출하였으며, 추출된 페이로드 데이터를 표 1의 두 번째 데이터 헤더에 기준 항, 8개의 Application Layer Payload Data 파일을 생성하였다. 그리고 같은 HTTP (Hypter Text Transfer Protocol) 를 사용하는 HTTP-Facebook, HTTP-Google, HTTP-Wikipedia, HTPP-Youtube의 경우 Web이라는 label로 통합하여, 최종적으로 총 5가지의 데이터 세트의 label을 결정하였다. 이를 통해 통합 과정을 거쳐 RDP, Skype, SSH, Bittorrent, Web의 Application Layer Payload Data 파일

	rdp	skype	ssh	bittorrent	Http-web
Total number of filtered flows	153,349	2,041	38,831	96,222	21,715
Total number of packets	6,875,730	3,015,153	17,910,988	207,306,103	47,136,213
Average number of packets in a flow	44	1477	461	2,154	2,483
Total packet size (GB)	131.3	12.0	321.7	2,170 (2.1TB)	698.4
Average size of all packets in a flow (MB)	0.9	6.0	8.5	23.7	39.9
Average packet size over all flows (KB)	20.0	4.2	18.8	11.3	15.9
Min packet size over all flows (B)	8.0	8.0	8.0	8.0	8.0
Max packet size over all flows (MB)	5.5	0.3	6.6	0.8	40.0

(그림 2) Application Layer 파일들의 통계 정보

을 완성하였다. 데이터 파일은 플로우마다 헤더 정보를 가지며, 페이로드는 패킷마다 '#' 이란 구분자로 구분하였다. 즉, 각개의 파일은 같은 label을 가진 플로우들의 집합이며, 패킷들은 페이로드만을 가지고 있고 페이로드는 패킷별로 구분되어진다. 또한 페이로드는 문자열로써, 한 문자의 크기가 4bit이다. 그림 2는 완성된 Application Layer Payload Data 파일의 통계 정보를 나타낸다.

2.2. Learning Data Generation

본 절에서는 RNN의 학습에 필요한 데이터 세트로 만들기 위하여 2.1 절에서 생성한 Application Layer Payload Data를 이용하여 학습용 데이터로 변환하는 과정을 설명한다. 학습을 위한 데이터 세트는 8750개의 플로우를 가진 'train data' 와 1250 개의 'validation data', 2500개의 'test data'를 갖는다. 각 train, validation, test 데이터 세트의 개수와 동일하게 각각의 label 데이터를 갖는다.

Application Layer Payload Data 파일의 각 Applicatione당 플로우는 RNN 학습을 위한 데이터로 변환하는 작업을 거치게 된다. 변환작업은 각 플로우의 Flow id, Start Time, End time을 나타내는 헤드 정보를 제거하고 Application Layer의 페이로드만을 나타내는 데이터 부분을 플로우 단위로 가져와 하나의 데이터 세트로 생성한다. RNN 모델에서 플로우 단위로 학습을 수행하기 위하여 한번에 Input 되는 데이터의 개수가 동일해야 한

다. 즉, 한 플로우가 가진 패킷의 크기가 동일해야 한다. 이를 위하여 사용자가 P 만큼의 한 플로우 당 패킷 개수를 지정 할 수 있다. 또한 플로우의 하나의 패킷은 사용자가 지정한 크기를 가지며 그 크기는 사용자가 N 만큼 설정 할 수 있고 1차원의 Array 크기를 갖게 된다. 한 패킷의 페이로드의 픽셀은 2^n bit의 크기를 가지며, 문자형 값을 부동소수 값으로 치환한 값을 갖게 된다. 즉, 한 플로우의 Data는 수식 (1) 과 같은 크기를 갖게 된다.

$$Flow\ data: N \times 2^n \times P \quad (1)$$

그림 3은 완성된 한 플로우의 패킷중 하나의 패킷의 페이로드를 Array 형태로 나타낸 것이며, 한 원소의 크기는 4bit이며, 부동소수 0.에서 15.사이의 값을 나타낸다. 또한 RNN 네트워크 구조의 특성상 모든 플로우는 패킷의 개수가 동일해야 한다. 왜냐하면, Sequence의 길이가 미리 설

```
[ 0.0   3.0   0.0   0.0   0.0   0.0   0.0  11.0
  0.0   6.0  13.0   0.0   0.0   0.0   0.0   0.0
  1.0   2.0   3.0   4.0   0.0   0.0   0.0   0.0
  0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
  0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
  0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
  0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
  0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
  0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
  0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0]
```

(그림 3) Flow's Packet의 페이로드 데이터

정되어 있기 때문에 해당 길이 만큼 플로우당 패킷의 개수가 동일해야 한다. 하지만, 모든 Application 플로우의 패킷의 개수가 동일 할 수 없기 때문에, 플로우당 패킷의 개수가 많은 경우 처음부터 기 설정한 N 만큼의 패킷만 플로우로 묶어서 학습을 한다. 하지만, 플로우의 패킷의 개수가 기 설정한 N의 개수보다 적은 경우 0.으로 패딩시킨 패킷을 생성하여 N의 개수를 맞추어 준다.

Train, Validation, Test label은 총 다섯 가지의 응용에 대한 label을 가지고 있으므로, 길이가 5인 one-hot vetor로써 표현할 수 있다. One-hot vector lable이란, 하나의 요소만 1인 값을 지니는 벡터로서, 1인 값의 인덱스가 응용 이름을 가리키는 label로 정의 할 수 있다. 표 2는 Train, Validation, Test label이 one-hot vector로 표현한 구조를 나타낸다.

3. RNN 모델 구조

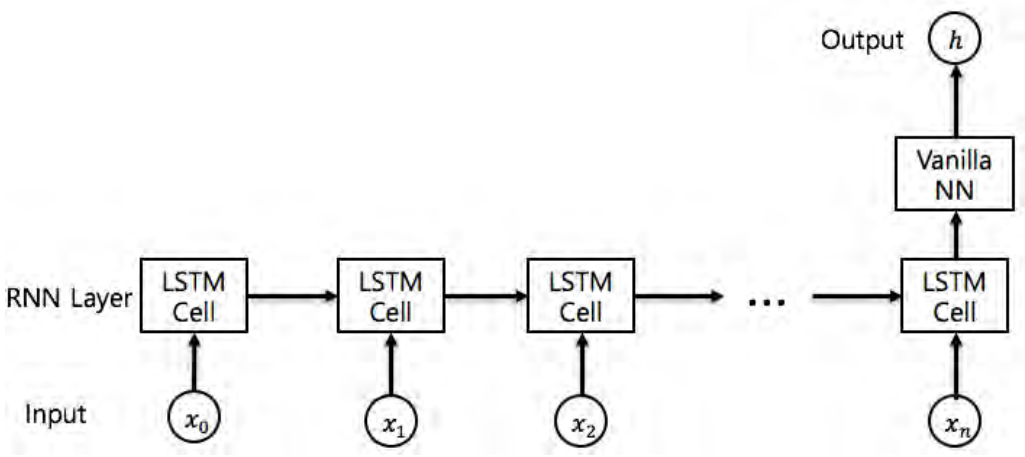
<표 2> Application One-hot Vector Label

Application	label
RDP	[1, 0, 0, 0, 0]
Skype	[0, 1, 0, 0, 0]
SSH	[0, 0, 1, 0, 0]
Bittorrent	[0, 0, 0, 1, 0]
Web	[0, 0, 0, 0, 1]

본 장에서는 제안하는 RNN을 이용한 플로우 기반 네트워크 트래픽 분류에서 사용하게 될 RNN 모델에 관한 설명을 한다. RNN은 다양한 자연어처리(NLP) 문제에 뛰어난 성능을 보이고 있으며, 특히 RNN은 순차적인 정보를 처리하는데 있어서 딥 러닝에서 많이 사용되는 CNN (Convolution Neural Network) 보다 순차적인 데이터 학습에 있어 뛰어난 성능을 보이고 있다. 따라서 패킷의 순차적인 정보를 담고 있는 플로우를 분석하고 해당 플로우의 분류를 하는데 있어 RNN을 통한 네트워크 트래픽 분류가 정확할 것이라 본다.

그림 4는 트래픽 분류에i 사용하는 RNN 모델의 학습 구조를 나타내며, 학습 모델은 Single Layer 로 구성되어 있다. RNN의 학습을 위한 시간적 스텝은 플로우 전체의 패킷들중 학습을 하고자 하는 패킷의 개수를 미리 설정한다. 시간적 스텝은 한 플로우를 Single Layer에서 RN으로 학습 시 순차적으로 들어가는 Input의 개수를 나타낸다. 따라서 학습 데이터 세트의 형태는 Train 데이터 세트의 경우 (플로우의 개수, 플로우당 패킷의 개수, 패킷당 페이로드 사이즈)로 나타내며, Label 데이터는 (플로우의 개수, 레이블의 개수)로 나타낸다. Test와 Validation의 데이터 세트도 위와 동일한 형태로 Input되어 RNN을 통해 학습이 수행된다.

기존 Vanilla RNN의 문제점은 Sequence의 길이가 길



(그림 4) RNN 모델 구조

어질 경우 장기 의존성 문제 (The Problem of Long-Term Dependencis)가 발생한다. 따라서 본 RNN 네트워크 구조에서는 장기 의존성 문제를 해결책으로 나온 LSTM (Long Short Term Memory Networks)를 이용한다 [6].

4. 실험 평가

본 장에서는 2장과 3장의 트래픽 데이터 처리 및 학습 데이터 생성 작업을 거친 데이터 세트를 RNN 모델에 Feeding 하여 훈련시켰다. 플로우 단위의 데이터는 플로우 당 패킷의 개수에 따라 얼마만큼의 학습을 할지 결정할 수 있다. 또한 RNN 훈련 시 한 셀에 들어가는 Input 데이터의 크기는 한 플로우의 한 패킷의 페이로드 사이즈와 동일하기 때문에 이 또한 플로우 기반의 분류에 영향을 줄 수 있다. 따라서, 본 논문에서 제안하는 플로우기반의 네트워크 트래픽 분류를 위하여 플로우당 패킷의 개수와 플로우내 한 패킷의 페이로드 사이즈를 다르게 하여 실험 평가를 진행하였다.

실험은 Ubuntu 14.04 LTS 환경에서 실험을 진행 하였고, RAM 32GB, NVIDIA GTX 1080Ti 를 사용하였다. 또한, TensorFlow 환경에서 Python을 이용하여 트래픽 분류를 위한 RNN 학습을 수행하였다.

실험 평가를 진행하기 위하여 한 플로우당 패킷의 개수를 10, 30, 60, 100개로 하였으며, 한 패킷 당 페이로드 사이즈를 40, 80, 160으로 설정하였다. 각 Label은 표 2의 내용을 토대로 Label 데이터를 생성하였으며, 각 플로우 개수는 Train, Validation, Test 별로 각각 8750개, 1250개, 2500개로 설정하였다. RNN 네트워크의 배치 사이즈는 1000 이며, RNN 의 Sequence Length는 플로우당 패킷의 개수와 동일하며, Hidden Size는 한 패킷 당 페이로드 사이즈와 동일하다. 최종 Output의 모양은 (플로우 개수, 5)의 형태로 출력된다.

		플로우당 패킷 개수							
		10		30		60		100	
		정확도 (%)	훈련 패킷 시퀀스 크기 (Bytes)	정확도 (%)	훈련 패킷 시퀀스 크기 (Bytes)	정확도 (%)	훈련 패킷 시퀀스 크기 (Bytes)	정확도 (%)	훈련 패킷 시퀀스 크기 (Bytes)
패킷당 페이로드 데이터 개수	40	96.26	40x4x10/8 =200	99.46	40x4x30/8 =600	99.72	40x4x60/8 =1200	99.74	40x4x100/8 =2000
	80	96.00	80x4x10/8 =400	99.74	80x4x30/8 =1200	99.75	80x4x60/8 =2400	99.82	80x4x100/8 =4000
	160	96.50	160x4x10/8 =800	99.74	160x4x30/8 =2400	99.88	160x4x60/8 =4800	99.85	160x4x100/8 =8000

(그림 5) 플로우 당 패킷의 개수와 페이로드 사이즈에 따른 Test Accuracy 결과

그림 5는 플로우당 패킷의 개수와 패킷의 페이로드 사이즈에 따른 Test Accuracy 결과를 나타낸다. 위 그림 5에서 보이는 바와 같이 플로우 당 패킷의 개수가 증가함에 따라 전체적인 정확도도 올라가는 것을 볼 수 있다. 또한, 한 패킷의 페이로드 사이즈가 커짐에 따라 정확도도 올라가는 것을 확인 할 수 있다. 따라서 본 논문에서 제안하는 RNN을 이용한 플로우 기반의 네트워크 트래픽 분류는 거의 99% 이상의 분류를 할 수 있으며, 데이터의 사이즈가 커짐에 따라 그 정확도가 더 증가하는 것을 확인하였다.

5. 결론

네트워크 트래픽 데이터를 플로우 단위로 하여 RNN 모델에 학습한 겨로가 그 정확도가 99% 이상에 달하였다. 이를 통하여 거의 100% 가깝게 분류가 가능 하다는 것을 확인하였다. 향후 연구에서는 실제 네트워크에 Deploy를 통하여 네트워크 데이터 트래픽의 분류에 대해 연구하며, 또한 이미 학습 되어진 패킷 이외에 새로운 패킷들이 실제 네트워크상에서 들어올 경우 분류 하는 방법에 대한 연구를 진행 할 것이다.

참고문헌

[1] 박진완, “통계 시그니쳐 기반의 응용 트래픽 분류,” 한국통신학회논문지 제 34권 제 11호, 2009.11, 1234-1244

[2] F. Risso, “Lightweight, Payload-Based Traffic Classification: An Experimental Evaluation,” Communications 2008 ICC’08, IEEE International Conference on. IEEE, 2008

[3] Yann LeCun, “Deep Learning,” Nature, International Weekly Journal of Science.

[4] Universitat Politècnica de Catalunya@Barcelon, Spain, HTTP://www.cba.upc.edu/monitoring/traffic-classification#is-our-ground-truth-for-traffic-classification-reliable-data set

[5] T. Mikolov, S. Kombrink, L. Burget, J. Černocký and S. Khudanpur, "Extensions of recurrent neural network language model," 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Prague, 2011, pp. 5528-5531.

[6] Sak, H & Senior, Andrew & Beaufays, F, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH. Jan, 2014, 338-342.