



국민대학교  
소프트웨어융합대학  
컴퓨터공학부

# 캡스톤 디자인 I

## 종합설계 프로젝트

프로젝트 명	SeMo(Security Monitoring Platform)
팀 명	Do Mo! (Do Monitoring!)
문서 제목	중간보고서

Version	1.2
Date	2009-04-30

팀원	전 하훈 (조장)
	김 성은
	최 운호
	최 현인
	허 윤서
지도교수	윤 명근 교수

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23


### CONFIDENTIALITY/SECURITY WARNING

이 문서에 포함되어 있는 정보는 국민대학교 소프트웨어 융합대학 소프트웨어학부 및 소프트웨어학부 개설 교과목 캡스톤 디자인I 수강 학생 중 프로젝트 "SeMo"를 수행하는 팀 "Do Mo!"의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 "Do Mo!"의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

## 문서 정보 / 수정 내역


<b>Filename</b>	중간보고서-SeMo.doc
<b>원안작성자</b>	전하훈, 김성은, 최운호, 최현인, 허윤서
<b>수정작업자</b>	전하훈, 김성은, 최운호, 최현인, 허윤서

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2020-04-15	전하훈	1.0	최초 작성	초안 작성
2020-04-17	김성은	1.1	내용 추가	데이터 분석 부분 추가
2020-04-17	허윤서	1.2	내용 추가	웹 부분 추가
2020-04-18	최운호	1.3	내용 수정	웹 부분 개발 내용 수정
2020-04-19	최현인	1.4	내용 추가	ELK 부분 추가
2020-04-20	전하훈	1.5	내용 추가	딥러닝 부분 개발 내용 및 향후계획 추가
2020-04-22	전원	1.6	내용 수정	오타자 및 변경된 사항 추가
2020-04-23	전하훈	1.7	최종 작성	최종 검토

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

## 목 차

1	프로젝트 목표	4
2	수행 내용 및 중간결과	5
2.1	계획서 상의 연구내용	5
2.1.1	데이터 분석	5.
2.1.2	ELK 스택	5.
2.1.3	웹	5.
2.1.4	딥러닝	7.
2.2	수행내용	8.
2.2.1	데이터 분석	8.
2.2.2	ELK	9.
2.2.3	웹	11.
2.2.4	딥러닝	12.
3	수정된 연구내용 및 추진 방향	28.
3.1	수정사항	28.
4	향후 추진계획	29.
4.1	향후 계획의 세부 내용	29.
4.1.1	데이터 분석	29.
4.1.2	ELK	29.
4.1.3	웹	29.
4.1.4	딥러닝	29.
5	고충 및 건의사항	30.

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

## 1 프로젝트 목표

프로젝트의 목표를 명확하게 기술한다.

Cisco 네트워크 트래픽 전망 조사에 따르면 앞으로 네트워크 트래픽의 양이 방대해 질 것이라고 전망하였고, 과학기술정보통신부에서 발표한 보안분야 매출 현황에서도 보안관제의 매출이 상승한 것으로 보아 앞으로의 네트워크 보안관제의 역할은 중요해진다. 그러나, 이러한 보안관제에 걸림돌인 "오탐(False Positive)"은 현저하게 많은 양의 데이터를 차지한다. KISTI (한국 과학 기술 정보 연구원)로부터 받은 IPS 로그데이터에서 정오탐 비율은 월 기준으로 많게는 1:9 적게는 3:7 (정탐:오탐) 비율이 나타나고 있다. 또한, 논문[1]에서 오탐의 존재는 보안관제사들이 정확한 분석을 하는데 방해를 주고, 오탐을 분석하는데 걸리는 시간이 대부분이라고 주장한다. 따라서 "DoMo!"는 이러한 오탐의 분류를 자동으로 할 수 있는 알고리즘을 개발하고, 이러한 정오탐 분석을 용이하게 해 줄 플랫폼을 구축한다. 해당 플랫폼은 보안관제 분야가 구축되지 않은 기업에 가이드라인이 될 수 있게 오픈소스 형태로 제공한다.

세부 목표

- 실시간으로 축적되는 IPS 로그에서 정오탐 분류를 실시한다.
- 정오탐 분류한 로그데이터를 사용자가 분석하기 쉽게 웹에서 시각화 한다.
- ELK 스택을 이용해 실시간 데이터 처리를 진행한다.
- ELK와 웹을 연동시켜 웹에서 시각화를 진행한다.
- 웹에서 사용자(보안관제사)가 자신들의 데이터에 맞게 재학습을 가능하게 한다.

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

## 2 수행 내용 및 중간결과

### 2.1 계획서 상의 연구내용

사용자의 Log 파일을 ELK와 웹에 연동시키면, ELK는 Log 파일을 실시간으로 처리하고, 실시간으로 처리되는 과정 중에 해당 로그파일 분석을 위해 학습된 모델로부터의 탐지 결과를 다시 ELK 스택에 입력해주고, 해당 결과를 받으면 ELK는 제시된 시각화 툴에 기반하여 웹을 통해 분석을 위한 데이터 시각화를 진행한다. 각 세부 연구 분야는 다음과 같다.

#### 2.1.1 데이터 분석

현재 IPS 로그 데이터로부터 정오탐 분류에 사용되고 있는 필드는 전문지식이 필요한 필드들이다. 해당 내용의 예로는, Source IP, Destination IP, Source Port, Destination Port, directionType 등과 같이 네트워크 전문 지식이 필요하고 해당 필드가 어떤 역할을 하는지 정확히 알고 있어야 정오탐 분류가 가능하다. 따라서 본 프로젝트에서는 정오탐 분류 자동화를 위해 다양한 정보가 담겨있고, 현재 사용되지 않고 있고, 일반적인 필드인 Payload분석 및 다양한 필드 분석을 진행한다.

#### 2.1.2 ELK 스택

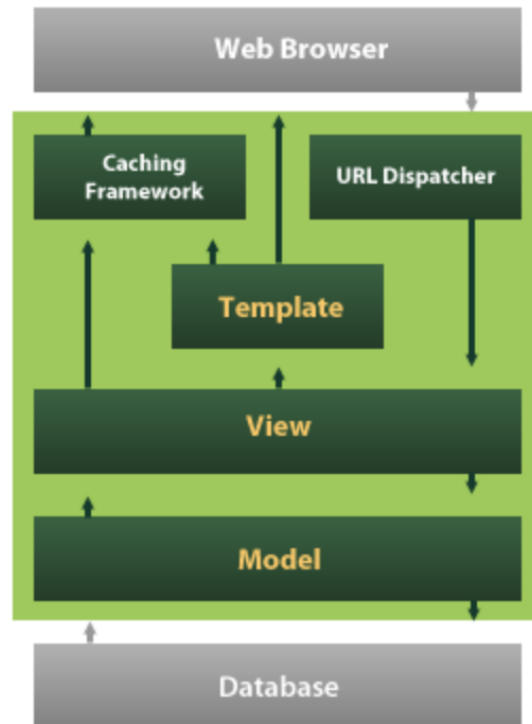
대용량 데이터의 실시간 처리를 위해서 ELK 스택을 사용한다. Logstash를 통해 실시간 데이터 또는 대용량 데이터를 Elasticsearch에 저장하고, Elasticsearch에 저장된 데이터를 Kibana 대시보드를 통해 효율적으로 시각화한다.

#### 2.1.3 웹

본 프로젝트에서는 시나리오를 위한 기본적인 웹 서버를 구현한다. ELK를 통해 시각화된 대시보드들은 장고의 url → view → model → template flow로 추출 된다. 추출한 대시 보드를 적절한 배치와 일관된 디자인을 통해 구조화 하여 사용자에게 제공한다.




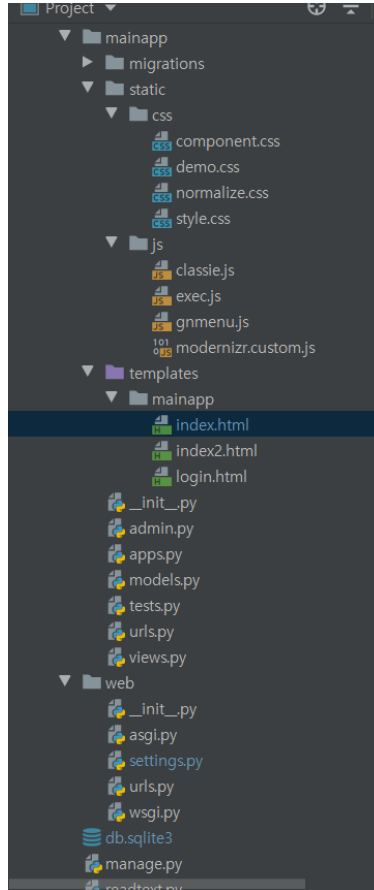
프로젝트 명	SeMo	
팀 명	Do Mo!	
Confidential Restricted	Version 1.7	2020-APR-23



[ 그림 1 ] 웹 서비스 구조

(출처 = <https://pythonhosted.org/MyTARDIS/architecture.html> )

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23



[ 그림 2 ] Pycharm 장고 파일 구성


## 2.1.4 딥러닝

### 2.1.4.1) Deep Learning 모델을 위한 Preprocessing

딥러닝 모델에서 데이터 정오탐 판별을 위해 이벤트 로그에서 Payload 필드만 가져와서 전처리한다. 전처리 된 데이터는 딥러닝 모델을 거치게 되고 모델은 floating vector 를 출력하게 된다.

### 2-2) Machine Learning 모델을 위한 Preprocessing

이벤트 로그에서 Payload와 무의미한 필드를 제외하여 전처리를 한다. 전처리 된 데이터는

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

기계학습을 거치게 되고 모델은 predicted Vector를 출력하게 된다.

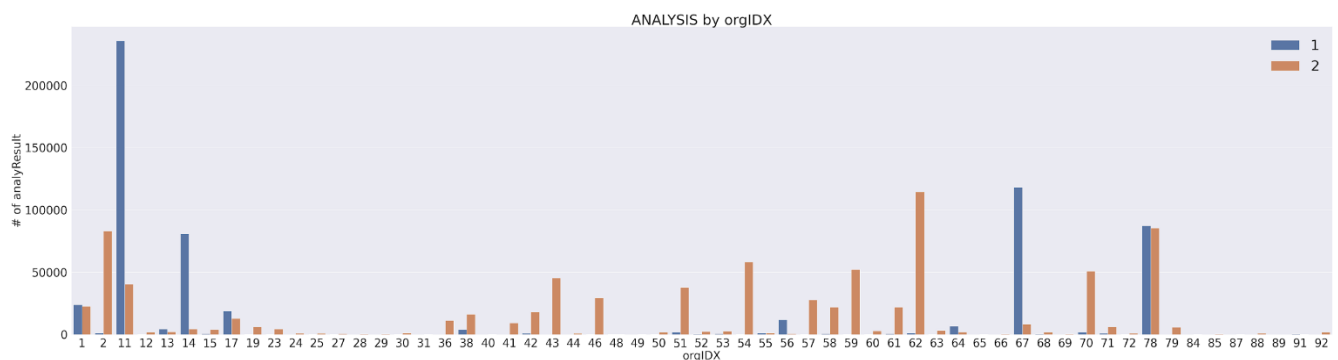
### 2-3) Ensemble 모델을 위한 Preprocessing

딥러닝 모델에서 나온 floating vector와 기계학습 모델에서 나온 predicted vector는 다시 한번 전처리(concated) 되어 앙상블 모델에 들어가게 된다. 데이터를 입력 받은 앙상블 모델은 해당 데이터가 정탐인지 오탐인지 정오탐 판별을 하게 된다.

정오탐 판별이 된 데이터 중 정탐인 데이터에 한하여 ELK Stack에 UPDATE 시킨 후 해당 이벤트와 관련된 데이터를 Kibana통해 웹에서 시각화 하여 보여지게 된다.

## 2.2 수행내용


### 2.2.1 데이터 분석



[ 그림 3 ] 기관별 정오탐 개수

각 기관에 대한 정탐과 오탐 판별 결과를 활용하여 기관에 대한 모델을 만들기 위해 그래프로



 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

각 기관에 대한 정탐과 오탐의 개수를 그림과 같이 나타냈다.

그림의 x축은 대상 기관의 인덱스를 표시했고 y축에는 정탐을 파란색으로, 오탐을 주황색으로 설정하여 그 개수를 표시했다. 그림을 보면 알 수 있듯이, 정탐과 오탐 데이터의 불균형이 심한 것을 알 수 있다.

따라서 기관에 대한 모델을 만드는 실험을 진행하지 않기로 결정했다.

## 2.2.2 ELK

ELK는 7.6.2 버전을 사용하였고, ELK 스택에 필요한 jvm은 Java SE Development Kit 11을 사용하였다.


개발 과정에서는 서버 컴퓨터를 SSH로 작업하고 있기에 서버의 IP 주소를 사용하여 Elasticsearch 데이터를 올리지만 배포되는 버전에서는 localhost로 작업가능 할 수 있도록 Logstash configuration 스켈레톤 파일을 제공한다.



```
1  input {
2    file {
3      path => "file path"
4      start_position => "beginning"
5      sincedb_path => "/dev/null"
6    }
7  }
8  }
9  filter {
10   csv {
11     separator => ","
12     columns => ["id","uid","stdrPort","atdate","autoFlag","sourcePort",
13               "payload","eventType","accidentType","destinationIP","packetSize",
14               "vfnStatus","detectEnd","attackType","detectStart","sourceIP",
15               "vfnUpdate","protocol","eventCount","destinationPort","batchID",
16               "detectName","metaType","detailResult","directionType","orgIDX",
17               "autoEmailSendFlag", "jumboPayloadFlag", "analyResult",
18               "doubtFlag","etcInfo"]
19   }
20   date{
21     match => ["atdate", "yyyy-MM-dd HH:mm:ss.SSS"]
22   }
23   mutate {remove_field => ["detectName"]}
24   mutate {convert => ["stdrPort", "float"]}
25   mutate {convert => ["sourcePort", "float"]}
26   mutate {convert => ["eventType", "float"]}
27   mutate {convert => ["accidentType", "float"]}
28   mutate {convert => ["attackType", "float"]}
29   mutate {convert => ["etcInfo", "float"]}
30 }
31 output {
32   elasticsearch {
33     hosts => ["http://localhost:9200"]
34     index => "IndexToElasticsearch"
35   }
36   stdout{}
37 }
```

[ 그림 4 ] logstash.conf

위 그림과 같이 현재 데이터는 Logstash pipeline을 거쳐서 데이터가 Elasticsearch에 올라가

 <div> <b>국민대학교</b>  <b>소프트웨어학부</b>  <b>캡스톤 디자인 I</b> </div>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

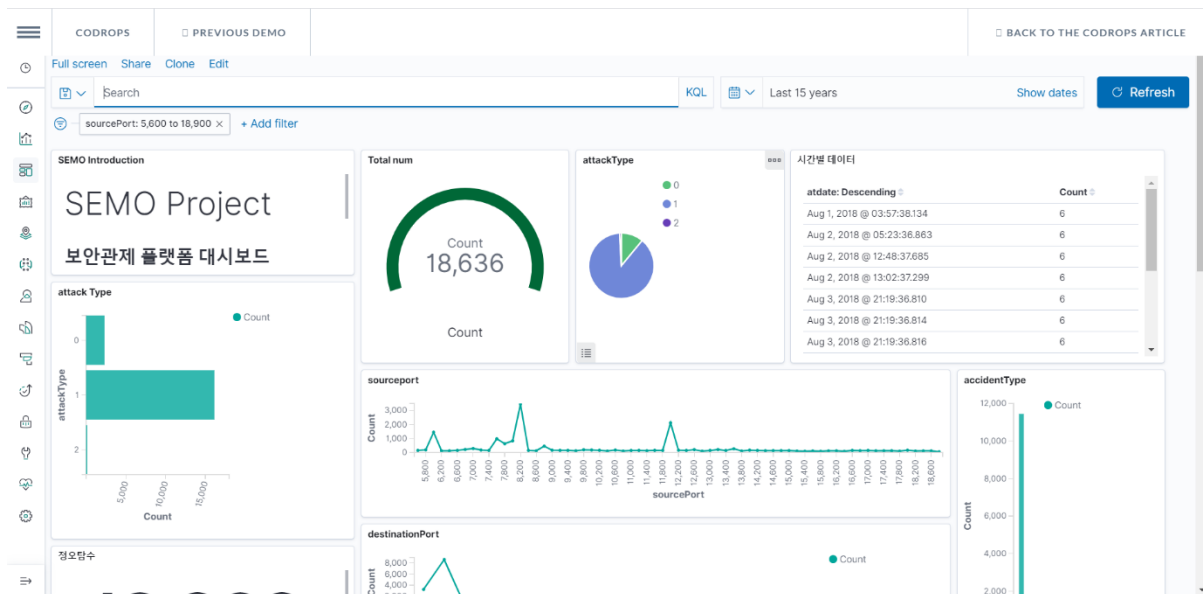
는데, pipeline 같은 경우 사용자가 자신의 데이터에 맞춰 변경하면 된다.

서버 컴퓨터에 ELK 구축을 완료하였고 데이터 업로드 및 실시간 데이터 업로드 테스트까지 마쳤다. 현재는 대용량 데이터를 대비하여 jvm Heap 사이즈 조절 등 유연한 Elasticsearch 환경을 설정하고 있다. 뿐만 아니라 사용자에게 보여줄 Kibana 대시보드의 구성을 지속적으로 고안하는 중이다.


### 2.2.3 웹

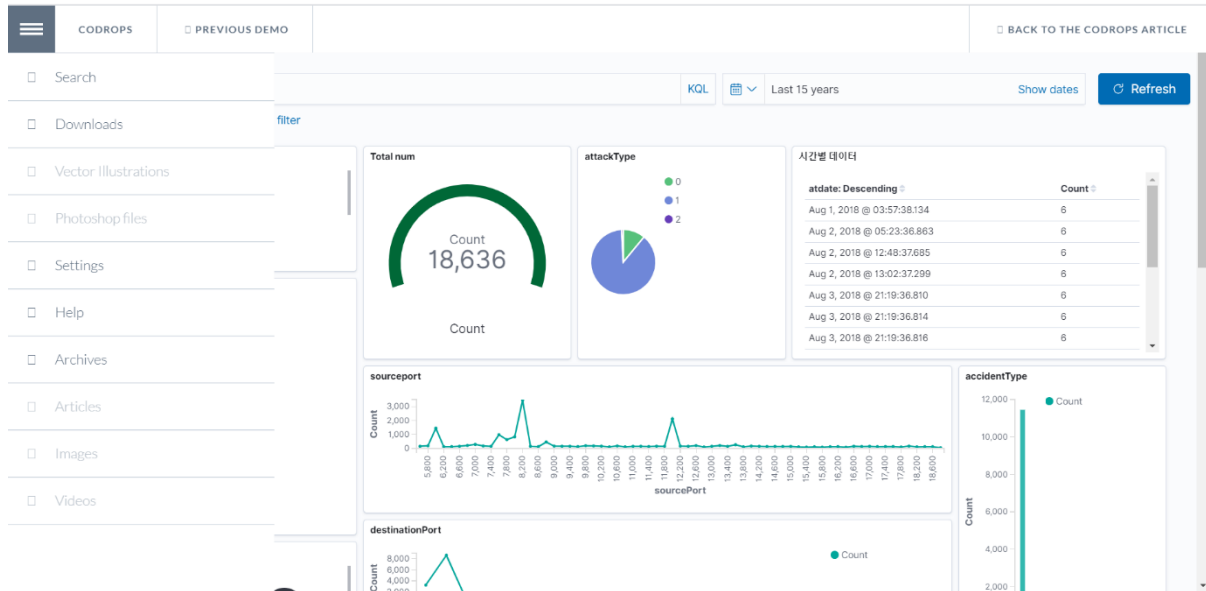
서버에 Django 3.0.4 설치 후 웹 서버 환경을 구축 했으며 웹 서버 동작 확인

사용자와 관리자의 접근을 분리하기 위한 로그인 페이지 및 데이터베이스 구축



[ 그림 5 ] 현재 웹 화면

 <div> 국민대학교  소프트웨어학부  캡스톤 디자인 I </div>	중간보고서		
	프로젝트 명	SeMo	
	팀 명	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23



[ 그림 6 ] 웹 메뉴바 클릭 시

현재 위 그림과 같이 url -> view -> model -> template 구조를 구축해 키바나 대시보드 임베딩까지 마친 상황이다. 사용자에게 최적화 된 서비스를 제공하기 위한 콘텐츠 별 프론트 초안 기능들을 개발 진행 중이며 모든 레이아웃은 부트스트랩 프레임워크를 이용하여 사용자의 브라우저 크기에 따라 변하는 반응형으로 제작하였다

## 2.2.4 딥러닝

정오탐 판별을 하기 위해 Tensorflow-Keras를 이용해 딥러닝 모델을 만든다. 페이로드를 바이트 스트림으로 간주하여 1dConv와 LSTM을 사용하여 모델 선정 실험을 진행했고, 임베딩의 중요성 실험하기위해 Raw값, One-hot encoding, Keras embedding layer로 실험을 진행한다.

여러 모델과 다양한 임베딩 기법을 사용하여 실험을 진행하고 테스트를 진행하여 각 성능을 확인해 테스트 결과를 기반으로 최종 모델 선정을 진행한다. 테스트 결과에 사용한 성능지표는 다음과 같다.

		실제 정답	
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

[ 그림 7 ] Confusion Matrix

- True Positive(TP) : 실제 정답을 정답이라고 예측 (정답)
- False Positive(FP) : 실제 오답을 정답이라고 예측 (오답)
- False Negative(FN) : 실제 정답을 오답이라고 예측 (오답)
- True Negative(TN) : 실제 오답을 오답이라고 예측 (정답)

1) 정확도 (Accuracy)


예측된 값이 실제 값과 얼마나 가까운지 나타내는 척도이다.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}}$$

2) 재현율 (Recall)

실제 정답 중에 예측한 정답을 나타내는 지표이다.

$$\text{Recall} = \frac{TP}{TP + FN}$$

 국민대학교 소프트웨어학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	SeMo	
	팀 명	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

### 3) 정밀도 (Precision)

예측한 정탐 중에 실제 정탐을 나타내는 지표이다.

$$Precision = \frac{TP}{TP + FP}$$

모든 실험에 사용된 데이터는 2018.08 ~ 2019.03 데이터로 구성했다. 해당 데이터 중에 2018.08 ~ 2019.01 데이터 1,443,437개를 학습에 사용했다. 검증 샘플은 2019.02 ~ 2019.03 데이터로 총 742,121개를 사용했다.

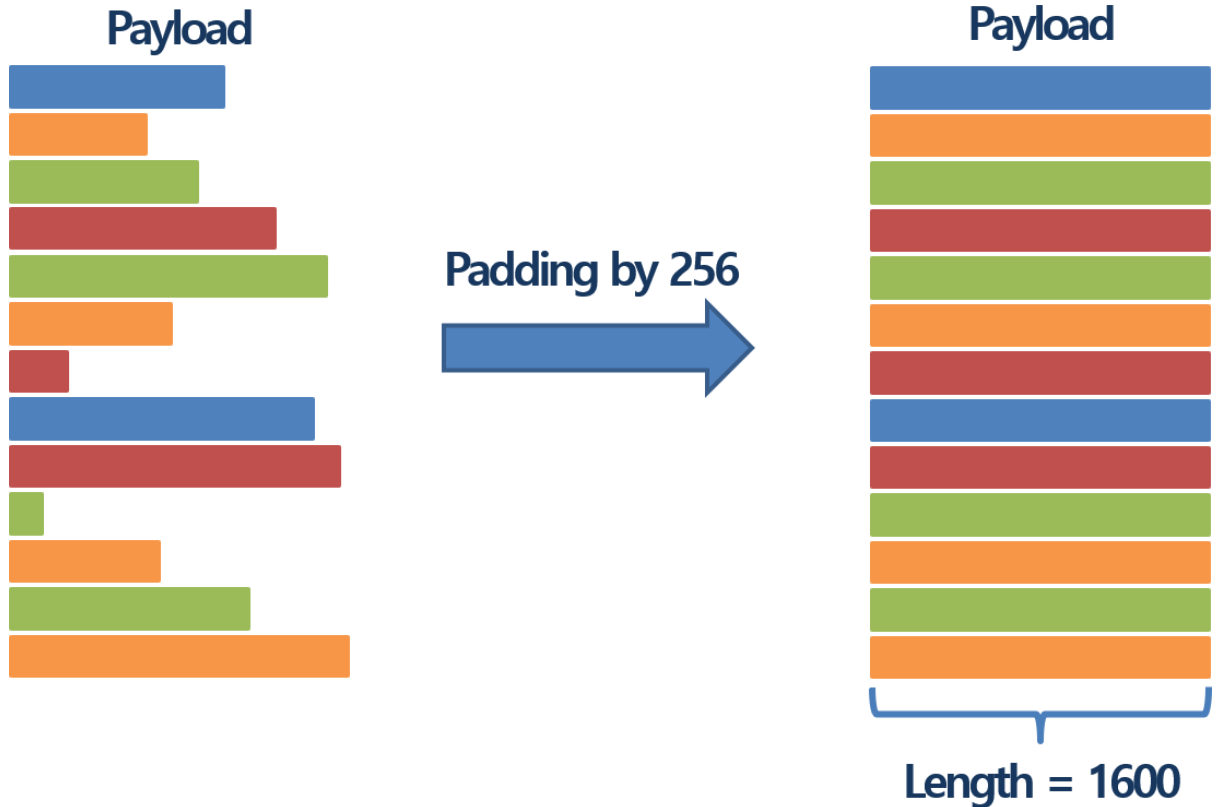
각 페이로드의 최대 1599 길이를 갖고, 각 바이트는 0~255 로 표현이 가능하다. 따라서 짧은 페이로드들을 훈련시키기 위해 의미 없는 값인 256으로 최대길이 1600으로 패딩 시키게 된다.

#### 2.2.4.2. 전처리

이벤트 로그데이터 파일에서 페이로드를 가져온 뒤 패딩을 거쳐 16진수를 10진수로 변환하는 전처리 과정을 갖게된다.



[ 그림 8 ] 전처리 구조



[ 그림 9 ] 패딩 방법

패딩은 페이로드의 최대 길이인 1600으로 패딩을 진행하고, 해당 값은 등장하지 않는 값인 256으로 패딩을 시킨다.


#### 2.2.4.3 모델 선정

##### 1) 1dConv vs LSTM

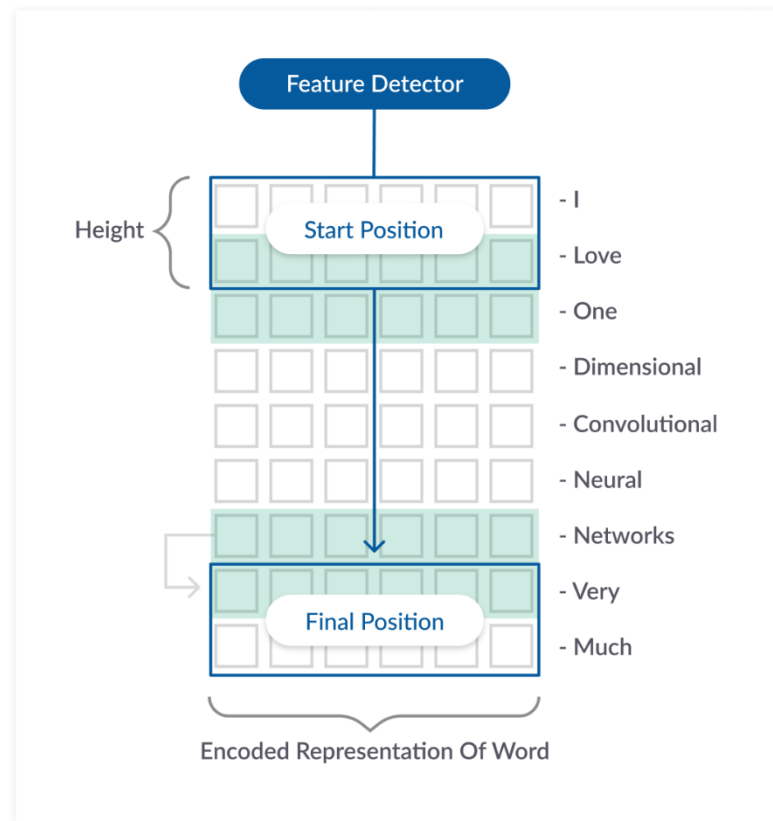
케라스의 1dConv레이어를 사용한 모델, LSTM을 사용한 모델의 성능을 비교하여, Convolution 과 RNN 모델 중에 적합한 모델을 선정한다. 각각 임베딩은 keras embedding layer로 진행했다.

##### 1-1) 1dConv

1dConv는 아래와 같이 작동한다. 하나의 페이로드를 바이트 스트림으로 간주하고

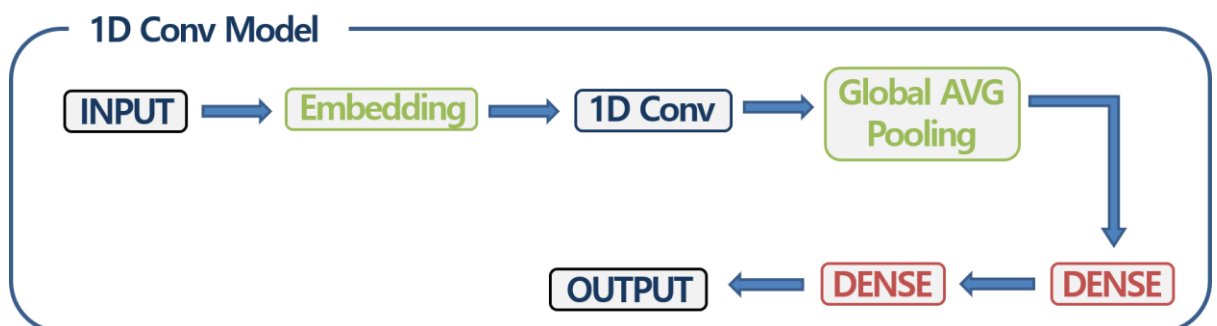
 국민대학교 소프트웨어학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	SeMo	
	팀 명	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

Kernel\_size를 3으로 설정하여 실험을 진행했다.




[ 그림 10 ] 1D Conv 작동 방식

[ 출처 : <https://missinglink.ai/guides/keras/keras-conv1d-working-1d-convolutional-neural-networks-keras/> ]



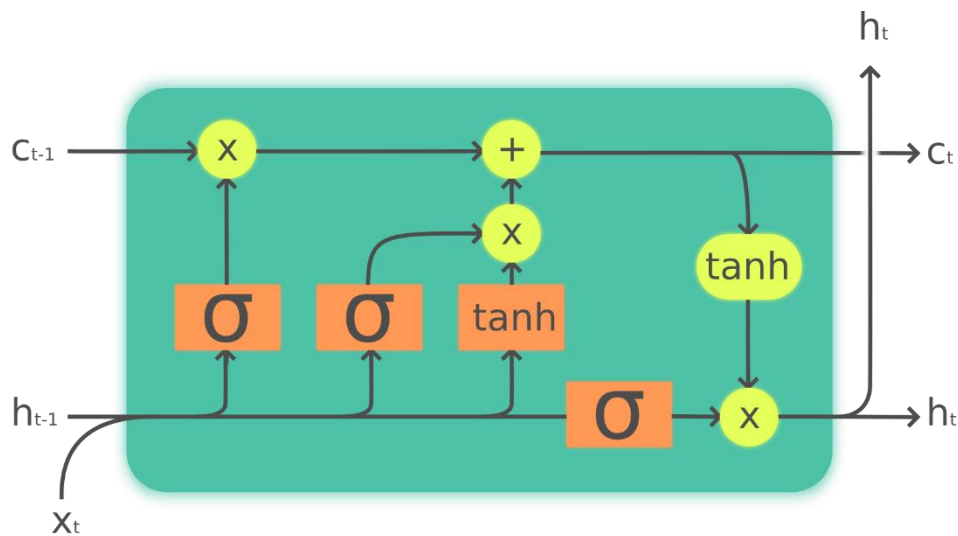
[ 그림 11 ] 1D Conv 모델 구조



 국민대학교 소프트웨어학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	SeMo	
	팀 명	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

## 1-2) LSTM

LSTM은 아래와 같이 이전에 판단한 결과들이 현재 판단하는 데이터에 영향을 주는 구조로 작동한다. 동일하게 Keras-embedding 레이어를 사용하여 임베딩 시킨 뒤 Keras에서 제공하는 LSTM의 기본 파라미터를 사용해 실험을 진행했다.

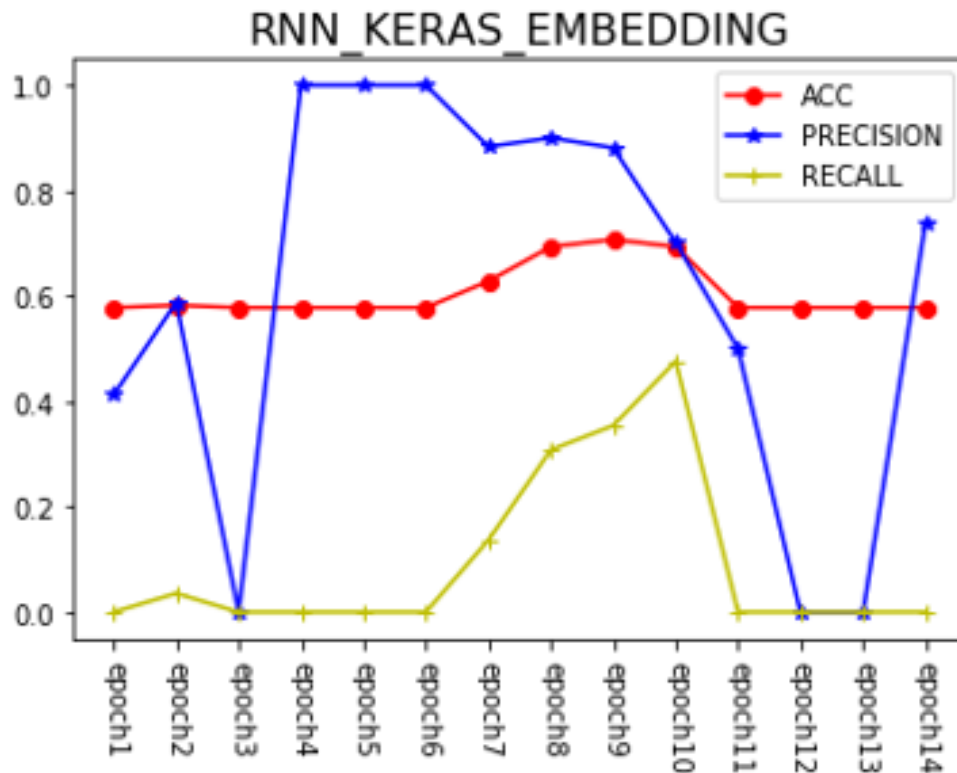


[ 그림 12 ] LSTM 작동 방식

[ 출처 : [https://en.wikipedia.org/wiki/Long\\_short-term\\_memory](https://en.wikipedia.org/wiki/Long_short-term_memory) ]



[ 그림 13 ] LSTM 모델 구조




[ 그림 14 ] LSTM 결과

LSTM으로 학습시켜본 결과, training accuracy가 약 0.63으로 학습이 진행이 안되는 것으로 판단하여, evaluate를 진행하지 않고 1dConv를 사용하기로 결정하였다.

## 2) 임베딩 선정

위 1) 에서 증명된 결과인 1dConv 모델을 구성하여 실험을 진행한다. embedding의 차이와 중요성을 알기 위한 실험으로 raw(0~1 정규화) , One-Hot, Keras-embedding 기법 중 적합한 기법을 정하여 모델을 구성한다.

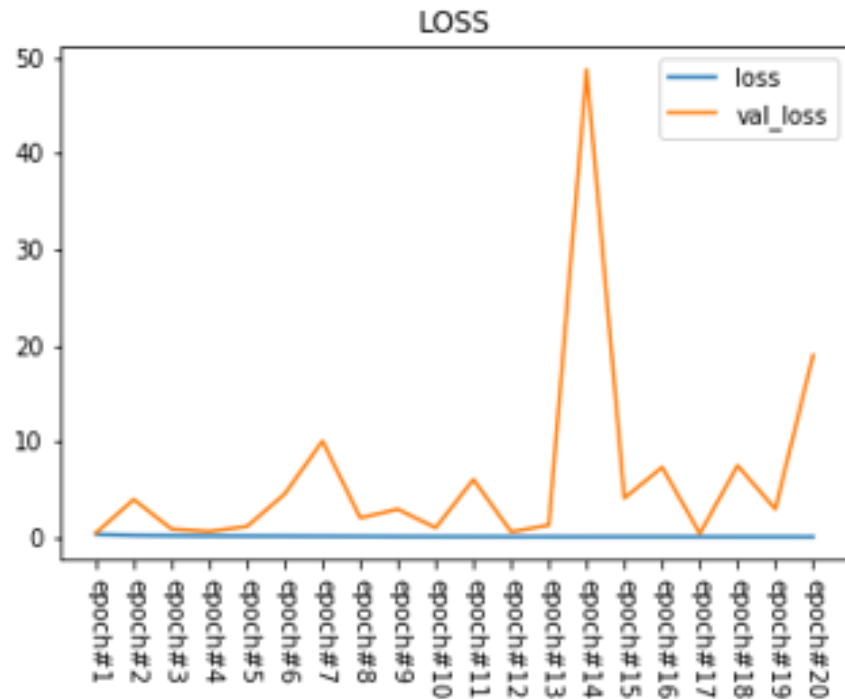
 국민대학교 소프트웨어학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	SeMo	
	팀 명	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

## 2-1) Raw

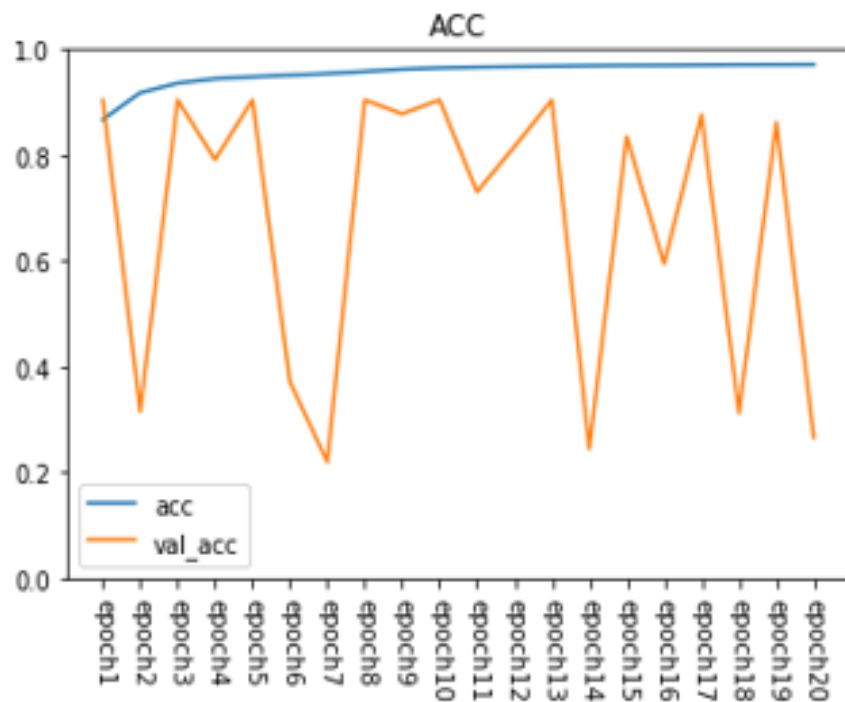
Raw 데이터의 페이로드는 각 1바이트로 표현가능하기 때문에 10진수로 바꾼뒤 0~1사이의 값으로 정규화 시켜서 훈련을 진행한다. 결국 각 페이로드들은  $1600 \times 1$  차원으로 인풋으로 들어가 학습에 사용된다.



[ 그림 15 ] 0~1 정규화 방법



[ 그림 16 ] 0~1 정규화 결과 LOSS 그래프



[ 그림 17 ] 0~1 정규화 결과 ACCURACY 그래프

Accuracy	Precision	Recall
0.8781	0.4360	0.9230

[ 표 1 ] 0~1 정규화 결과 표

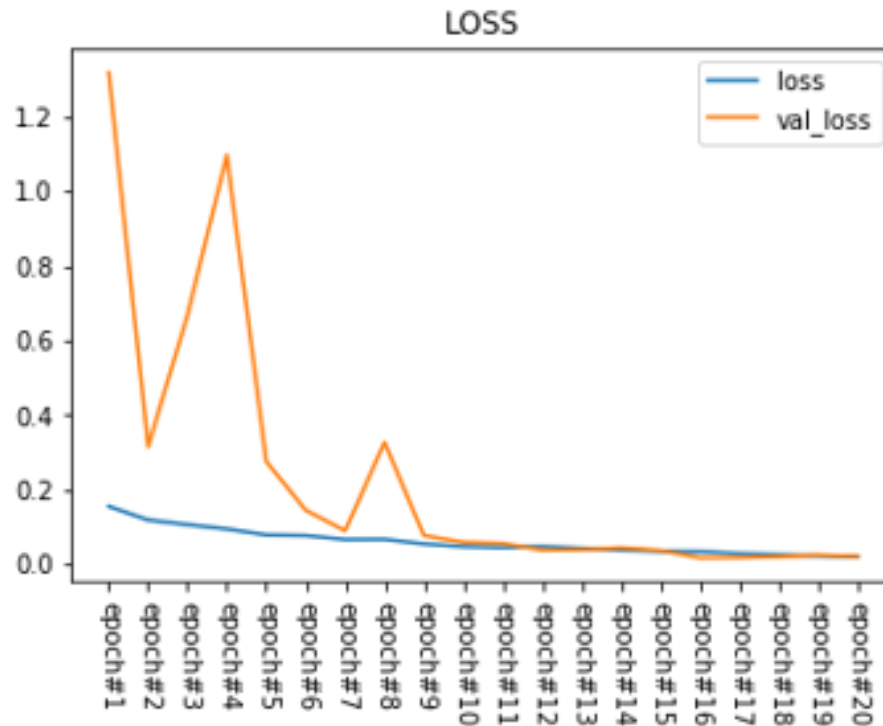
그림 14, 그림 15, 표 1에서 볼 수 있듯이 단순히 페이로드의 값을 그대로 넣어주는 것은 학습이 되지 않는다.

## 2-2) One Hot

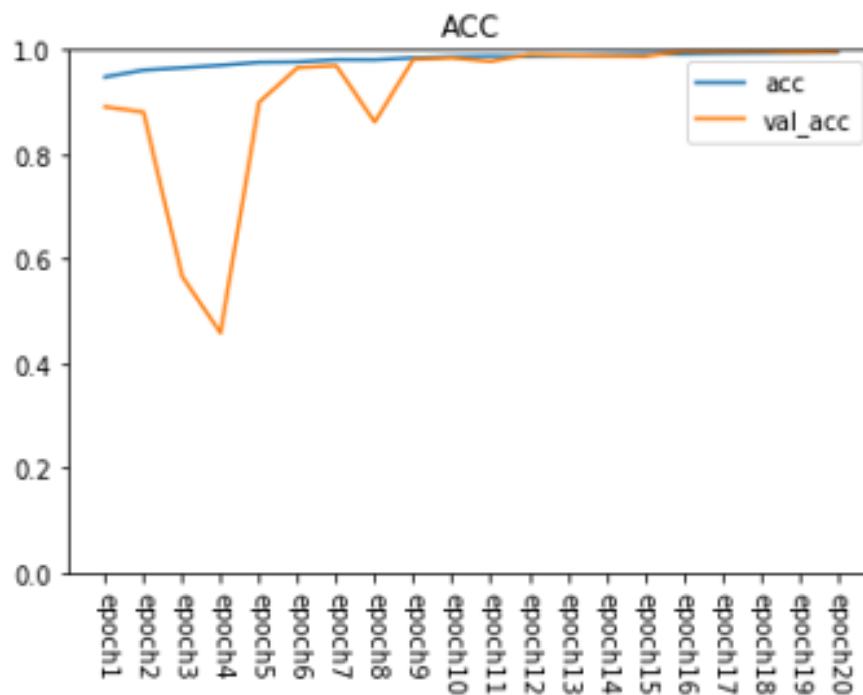
One-hot encoding 기법은 각 값에 대해 유니크한 공간을 할당해주는 기법이다. 아래 그림과 같이 5개의 단어들은 5차원 벡터로 들어가게 되고 각 단어를 의미하는 공간에 0, 1로 표현된다. 결국 각 페이로드들은 1600\*257 차원(0~256)으로 인풋으로 들어가 학습에 사용된다.

Index	Job	One hot encoded data					
1	Police	[	1	0	0	0	0]
2	Doctor	[	0	1	0	0	0]
3	Student	[	0	0	1	0	0]
4	Teacher	[	0	0	0	1	0]
5	Driver	[	0	0	0	0	1]


[ 그림 18 ] One-hot 인코딩 기법



[ 그림 19 ] One-Hot 인코딩 결과 LOSS 그래프




[ 그림 20 ] One-Hot 인코딩 결과 ACCURACY 그래프

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

Accuracy	Precision	Recall
0.9967	0.9756	0.9912

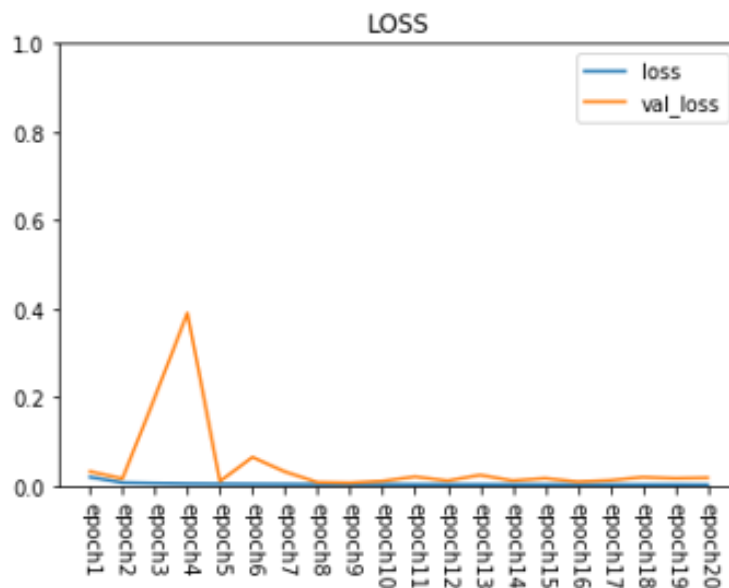
[ 표 2] One-Hot 인코딩 결과 표

그림 17, 그림 18, 표 2에서 볼 수 있듯이 페이로드의 값을 그냥 넣어주는 것 보다는 인코딩 또는 임베딩의 필요성을 알 수 있다. 그러나 One-Hot 인코딩 기법은 1600 x 257의 차원을 갖기 때문에 비교적 무거운 구조를 갖는다.

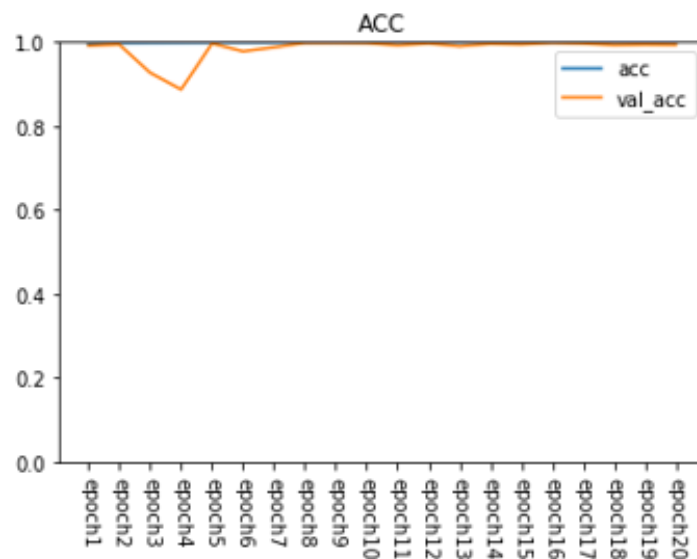
 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

### 3-3) keras-embedding

keras의 임베딩 레이어를 사용하여 각 값을 의미 있는 값으로 임베딩 시켜주는 레이어이다. 따라서 각 페이로드를 1600\*8차원으로 변환한 뒤 학습을 진행한다. 해당 모델은 다음과 같은 구조를 갖는다.




[ 그림 21 ] Keras embedding 결과 LOSS 그래프



[ 그림 22 ] Keras embedding 결과 ACCURACY 그래프



 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

Accuracy	Precision	Recall
0.9980	0.9839	0.9960

[ 표 3 ] Keras embedding 결과표

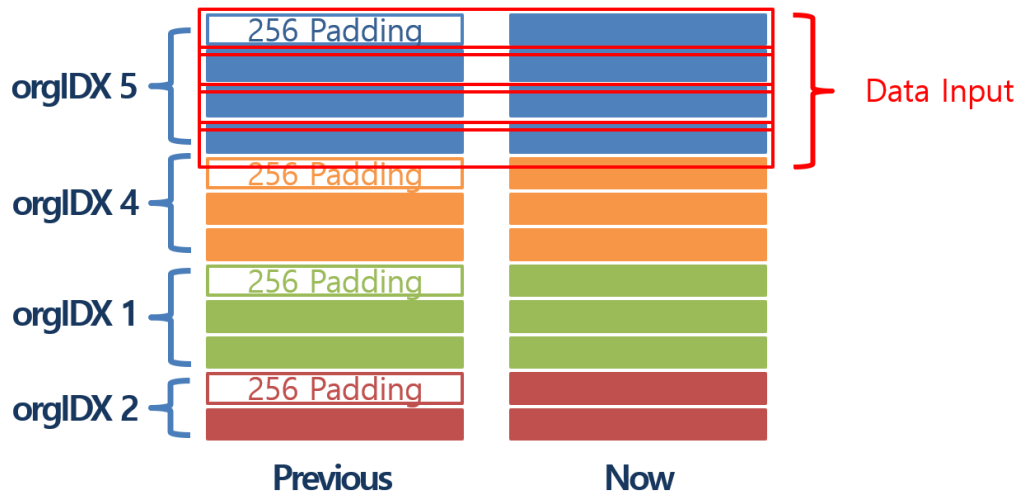
그림 19, 그림 20, 표 3에서 볼 수 있듯이 Keras embedding layer를 사용하는 것이 제일 좋은 결과를 보여주고 있다. 그리고, One-Hot 인코딩 기법보다 가벼운 구조를 가져서 보다 빠르게 학습이 진행된다. 따라서 임베딩 기법은 Keras의 embedding layer를 사용할 것이다.

### 3) pre-now Model

IPS 특성상 각 기관에 대한 데이터의 양, 특성, 룰셋의 최적화 여부 등 차이점을 보이기 때문에 데이터를 각 기관 별로 모은 뒤 실험을 진행하고, 기관 별로 데이터를 모을 경우 직전에 페이로드가 현재 페이로드와 연관성이 있다고 생각하여, 모델 구조를 직전 데이터가 현재 데이터에 영향을 주도록 바꾸고 그에 맞게 전처리 방법도 변경한다.

#### 3-1) pre-now Model 전처리

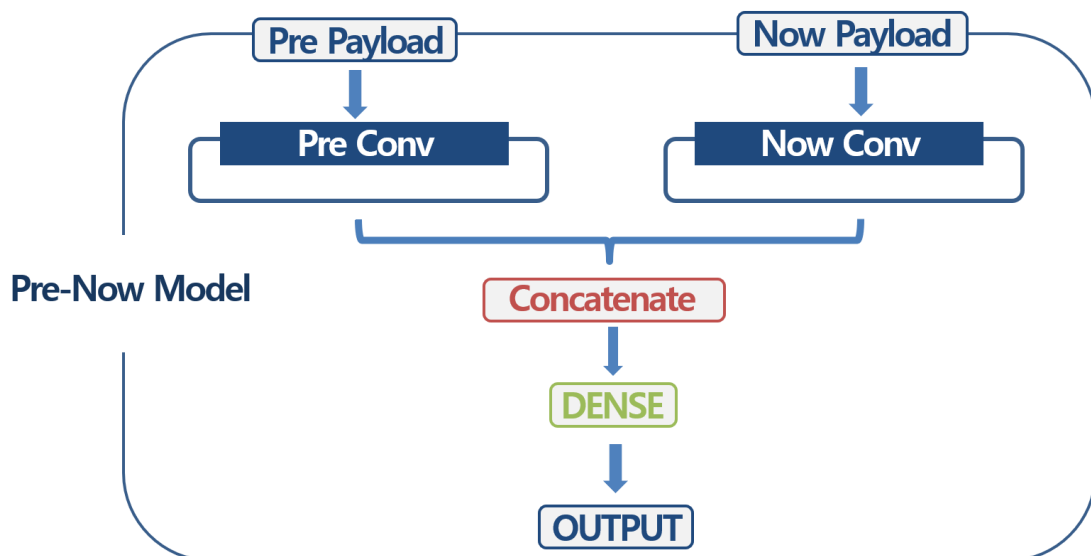
전처리로는 먼저, orgIDX (대상기관 인덱스) 별로 그룹화하여, 시간순으로 정렬하고, 정렬한 데이터의 직전 페이로드를 Previous 데이터로 만든다. 만들 때, 제일 앞에 나온 페이로드의 경우 256으로 패딩 한 값이 previous로 들어가게 된다.



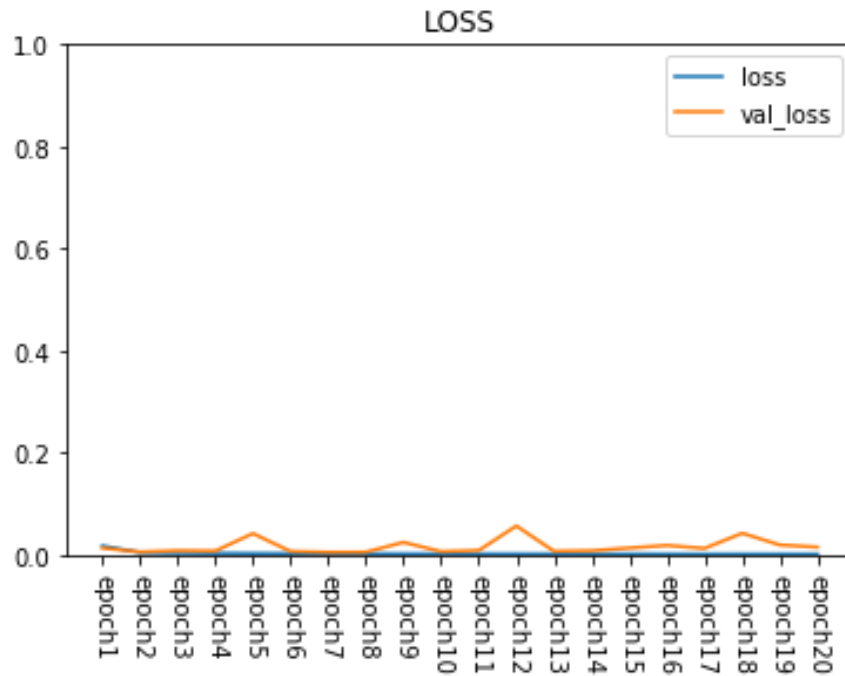
[ 그림 23 ] Pre-Now 전처리 방법

### 3-2) pre-now Model 구조

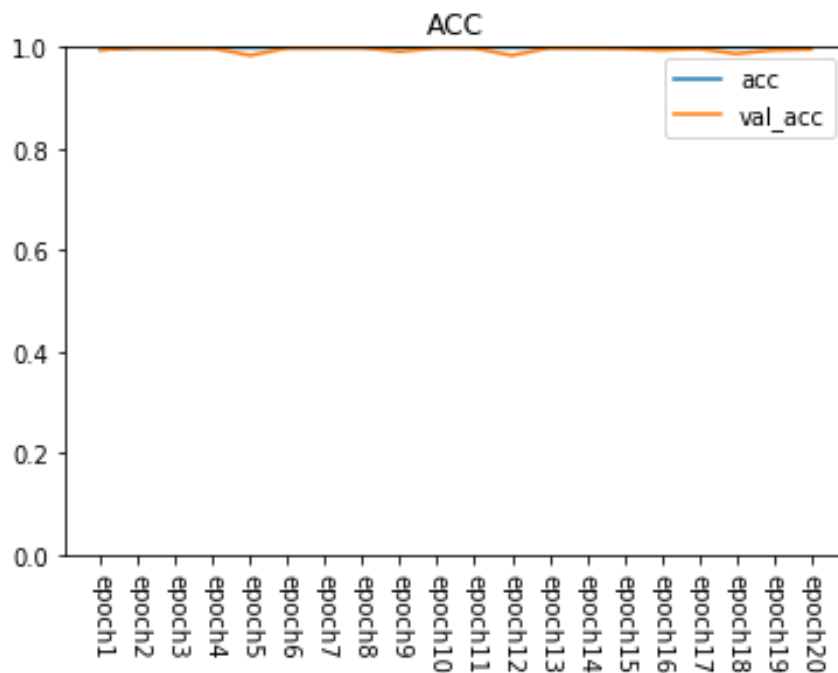
앞서 보여준 Keras embedding을 사용한 1D Conv를 사용하여 이전페이로드와 현재 페이로드를 데이터 인풋으로 넣어준다. 각각 Conv 레이어가 끝난 후 Concatenate되고 마지막에 Dense레이어를 거쳐 결과값을 출력하게 된다.




[ 그림 24 ] Pre-Now 모델 구조



[ 그림 25 ] Now-Pay 모델 결과 LOSS 그래프



[ 그림 26 ] Now-Pay 모델 결과 ACCURACY 그래프

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

Accuracy	Precision	Recall
0.9987	0.9904	0.9958

[ 표 4 ] Now-Pay 모델 결과표

그림 25, 그림 26, 표 4와 같이 현재 제일 좋은 성능을 보이고 있으며, Loss와 Accuracy가 보다 빠르게 수렴하는 것을 볼 수 있다. 따라서 현재 Now-Pay 모델을 안정화 및 검토해 나갈 것이다.

### 3 수정된 연구내용 및 추진 방향

#### 3.1 수정사항

##### 3.1.1 대시보드 분석내용 PDF 저장 기능 제외

키바나 베이직 무료 라이선스에서는 제공되지 않으나 상위버전 라이선스에서는 분석내용 PDF 전환 기능을 제공한다.

##### 3.1.2 대시보드에서 데이터 다운로드 기능 제외

키바나에서 데이터를 기간별 혹은 열 별로 csv형식으로 다운로드 기능을 제공한다. 또한 데이터 다운로드시 데이터의 중복 문제가 발생한다.

##### 3.1.3 딥러닝 모델 양상불, 머신러닝 모델 제외

본 프로젝트에서 방향성에 어긋나는 개발로 판단하여 제외하였습니다.

 <b>국민대학교</b> <b>소프트웨어학부</b> <b>캡스톤 디자인 I</b>	<b>중간보고서</b>		
	<b>프로젝트 명</b>	SeMo	
	<b>팀 명</b>	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

## 4 향후 추진계획

### 4.1 향후 계획의 세부 내용

#### 4.1.1 데이터 분석

올해 KISTI 과학기술 사이버 안전센터로부터 새로 받아온 데이터에 변경사항이 없는지 데이터 분석을 진행할 예정이다.

#### 4.1.2 ELK


효율적인 시각화를 위하여 Kibana 대시보드 구성을 고안하여 최종 버전을 배포하기 전까지 수정을 거듭할 것이다. 뿐만 아니라 사용자가 배포된 버전을 사용할 때 ELK 스택을 손쉽게 구현할 수 있도록 세부 구현과정을 문서화하여 제공할 것이다.

#### 4.1.3 웹

사용자의 요구사항을 고려하며 콘텐츠를 고안하고 요청에 따른 최적화 서비스를 제공을 위해 적절한 배치와 일관된 디자인 고려할 것이다.

#### 4.1.4 답러닝

Pre-Now Model을 더욱 개선할 것이며 데이터 분석에서 추가 데이터에 변경사항이 없을 경우 테스트 데이터로 활용해 최종 모델을 정할 것이다. 또한, 사용자가 편히 사용할 수 있도록, 모델에 모듈화를 진행할 예정이다. ELK 스택이 데이터를 읽어 드릴 수 있도록 후처리 기능을 개발할 예정이다. 시간이 된다면, 새로운 데이터에 대비하기 위해 주기적 재학습 기능을 추가할 예정이다.

 국민대학교 소프트웨어학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	SeMo	
	팀 명	Do Mo!	
	Confidential Restricted	Version 1.7	2020-APR-23

## 5 고충 및 건의사항

없습니다.