



**a security insight**

캡스톤 디자인 5조 어시스트

01

프로젝트 목표

02

진행 상황

03

계획 및 제한요소

01

프로젝트 목표

02

진행 상황

03

계획 및 제한요소

## asi - 핵심 아이디어

## 01 파일 입력

## 02 disassemble

## 03 RNN

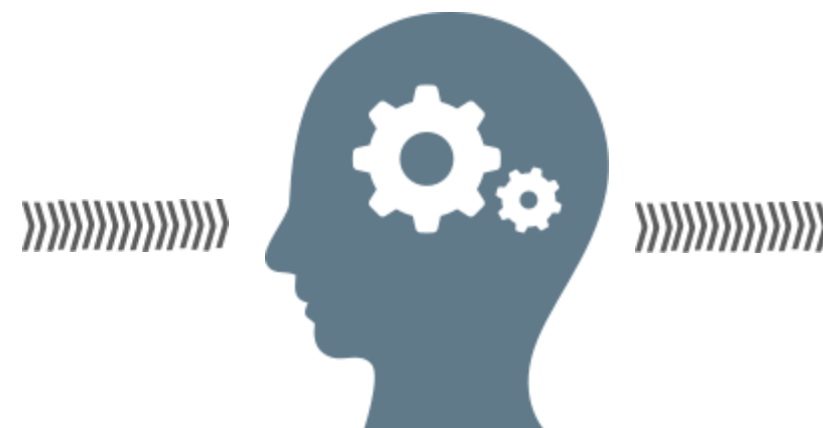
## 04 악성 행위 예측



```

10001180: 55      push %ebp
10001181: 8b ec   mov %esp,%ebp
10001183: 6a ff   push $0xffffffff
10001185: 68 21 80 00 10 push $0x10008021
1000118a: 64 a1 00 00 00 00 mov %fs:0x0,%eax
10001190: 50      push %eax
10001191: 83 ec 08 sub $0x8,%esp
10001194: a1 00 40 02 10 mov 0x10024000,%eax
10001199: 33 c5   xor %ebp,%eax
1000119b: 89 45 f0 mov %eax,-0x10(%ebp)
1000119e: 53      push %ebx
1000119f: 56      push %esi
100011a0: 57      push %edi
100011a1: 50      push %eax
100011a2: 8d 45 f4 lea -0xc(%ebp),%eax
100011a5: 64 a3 00 00 00 00 mov %eax,%fs:0x0
100011ab: 6a 14   push $0x14
100011ad: 6a 0c   push $0xc
100011af: ff 15 44 b2 01 10 call *0x1001b244
100011b5: 83 c4 08 add $0x8,%esp
100011b8: 89 45 ec mov %eax,-0x14(%ebp)
100011bb: 8b c8   mov %eax,%ecx
100011bd: c7 45 fc 00 00 00 00 movl $0x0,-0x4(%ebp)
100011c4: e8 c7 04 00 00 call 0x10001690

```



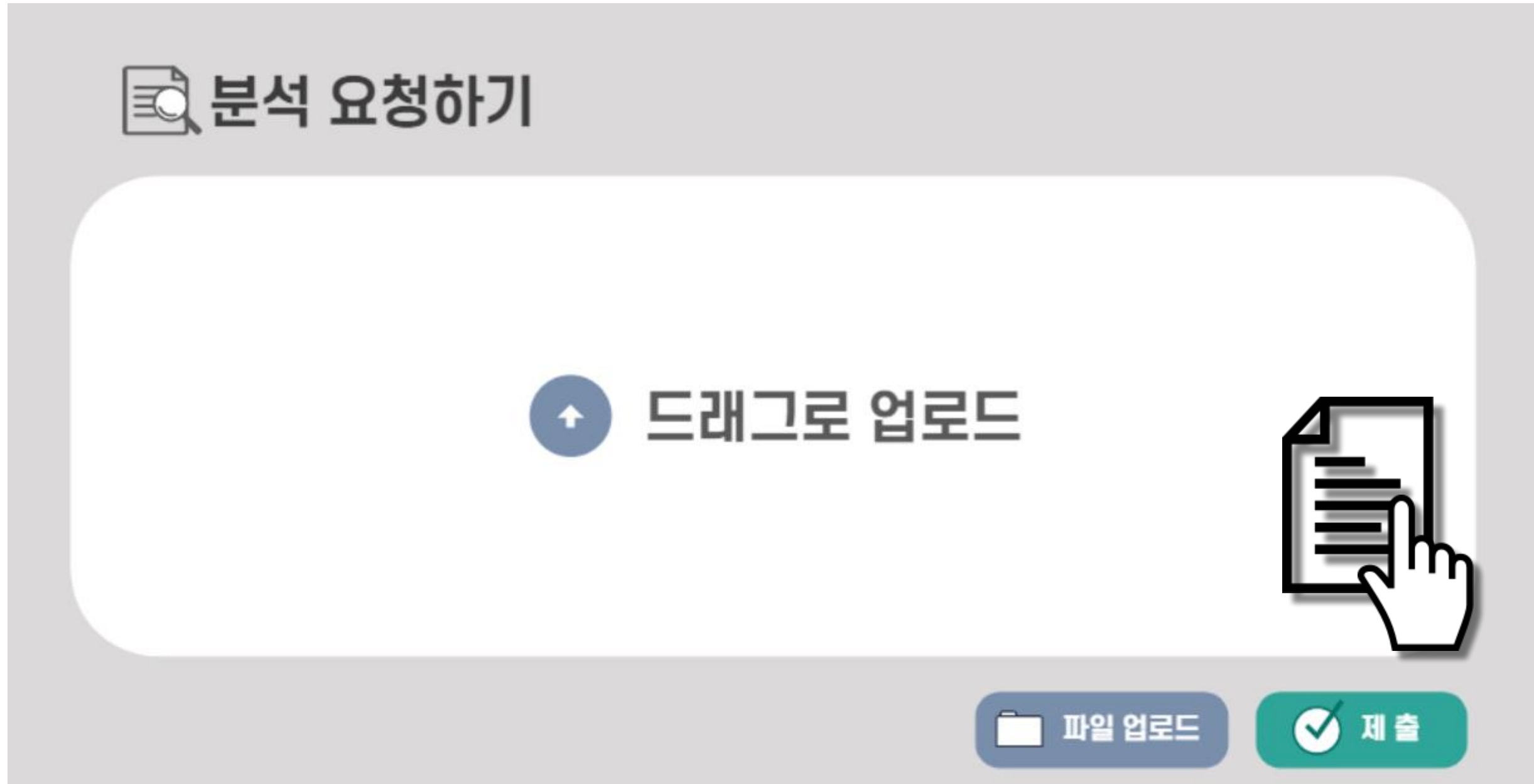
```

10001180: 55      push %ebp
10001181: 8b ec   mov %esp,%ebp
10001183: 6a ff   push $0xffffffff
10001185: 68 21 80 00 10 push $0x10008021
1000118a: 64 a1 00 00 00 00 mov %fs:0x0,%eax
10001190: 50      push %eax
10001191: 83 ec 08 sub $0x8,%esp
10001194: a1 00 40 02 10 mov 0x10024000,%eax
10001199: 33 c5   xor %ebp,%eax
1000119b: 89 45 f0 mov %eax,-0x10(%ebp)
1000119e: 53      push %ebx
1000119f: 56      push %esi
100011a0: 57      push %edi
100011a1: 50      push %eax
100011a2: 8d 45 f4 lea -0xc(%ebp),%eax
100011a5: 64 a3 00 00 00 00 mov %eax,%fs:0x0
100011ab: 6a 14   push $0x14
100011ad: 6a 0c   push $0xc
100011af: ff 15 44 b2 01 10 call *0x1001b244
100011b5: 83 c4 08 add $0x8,%esp
100011b8: 89 45 ec mov %eax,-0x14(%ebp)
100011bb: 8b c8   mov %eax,%ecx
100011bd: c7 45 fc 00 00 00 00 movl $0x0,-0x4(%ebp)
100011c4: e8 c7 04 00 00 call 0x10001690

```

악성행위 의심 영역 하이라이팅

## asi - 웹 서비스



<파일 업로드 시각화 안>

## asi - 웹 서비스

dasjfiwf.exe 1034 lines

17	6a ff	push	\$0xffffffff
18	68 21 80 00 10	push	\$0x10008021
19	64 a1 00 00 00 00	mov	%fs:0x0,%eax
20	50	push	%eax
21	83 ec 08	sub	\$0x8,%esp

< 2 / 12 > 목록으로 전체보기

<분석 결과 시각화 안>

01

프로젝트 목표

02

진행 상황

03

계획 및 제한요소

## 데이터 수집

**kaggle**  
microsoft  
malware prediction

바이트 파일 : **10,868개**  
어셈블리 파일 : **21,738개**  
총 **21,738개**

**KISA**  
Korea Internet & Security Agency  
정보보호 R&D  
데이터챌린지

정상 파일 : **10,000개**  
악성 파일 : **10,000개**  
**2017년 ~ 2019년**



자체 개발  
웹 크롤러

시스템 DLL 파일  
STEAM사 게임 인스톨러



## 니모닉 추출



**disassemble**  
IDA

10001180:	55	push	%ebp
10001181:	8b ec	mov	%esp,%ebp
10001183:	6a ff	push	\$0xffffffff
10001185:	68 21 80 00 10	push	\$0x10008021
1000118a:	64 a1 00 00 00 00	mov	%fs:0x0,%eax
10001190:	50	push	%eax
10001191:	83 ec 08	sub	\$0x8,%esp
10001194:	a1 00 40 02 10	mov	0x10024000,%eax
10001199:	33 c5	xor	%ebp,%eax
1000119b:	89 45 f0	mov	%eax,-0x10(%ebp)
1000119e:	53	push	%ebx
1000119f:	56	push	%esi
100011a0:	57	push	%edi
100011a1:	50	push	%eax
100011a2:	8d 45 f4	lea	-0xc(%ebp),%eax
100011a5:	64 a3 00 00 00 00	mov	%eax,%fs:0x0
100011ab:	6a 14	push	\$0x14
100011ad:	6a 0c	push	\$0xc
100011af:	ff 15 44 b2 01 10	call	*0x1001b244
100011b5:	83 c4 08	add	\$0x8,%esp
100011b8:	89 45 ec	mov	%eax,-0x14(%ebp)
100011bb:	8b c8	mov	%eax,%ecx
100011bd:	c7 45 fc 00 00 00 00	movl	\$0x0,-0x4(%ebp)
100011c4:	e8 c7 04 00 00	call	0x10001690

**parser**  
python, IDA

push  
mov  
push  
push  
mov  
push  
sub  
mov  
xor  
mov  
push  
push  
push  
push  
lea  
mov  
push  
push  
call  
add  
mov  
mov  
movl  
call

**File**

**Assembly code**

**Mnemonic**

## 단어 임베딩

### Word2Vec

유사한 단어를  
가까운 공간에 사상



### CBOW

주변 단어로  
중간 단어 예측



### SkipGram

중간 단어로  
주변 단어 예측



### *fast*Text

단어의 부분이 일치하면  
유사하다고 판단

## 단어 임베딩

| **실험 조건**    **gensim** 라이브러리 사용

- 01 윈도우 크기 : 10
- 02 최소 단어 수 : 50
- 03 에폭 : 10
- 04 학습률 : 0.002
- 05 특징 벡터 차원 : 8/16/32/64/128

```
mov jmp add pop push
```

## 단어 임베딩

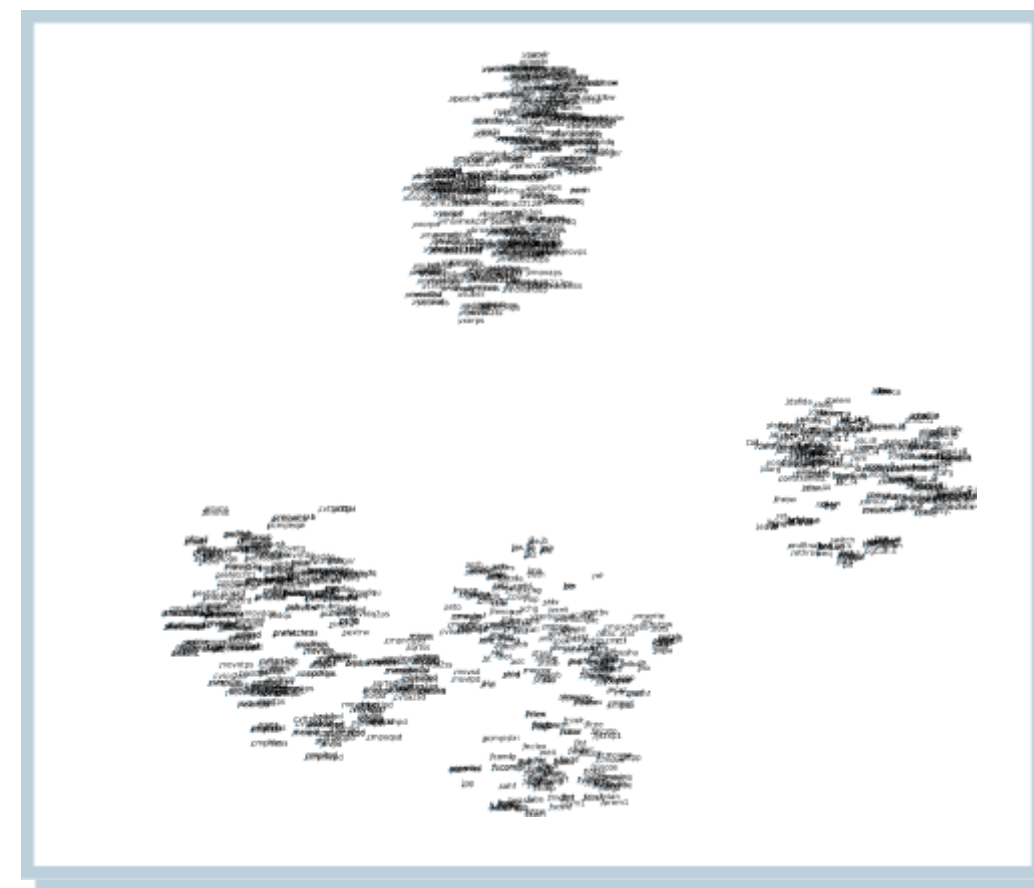
### | 실험 결과

# SkipGram

특징 벡터 차원 : 8/16/**32/64**/128

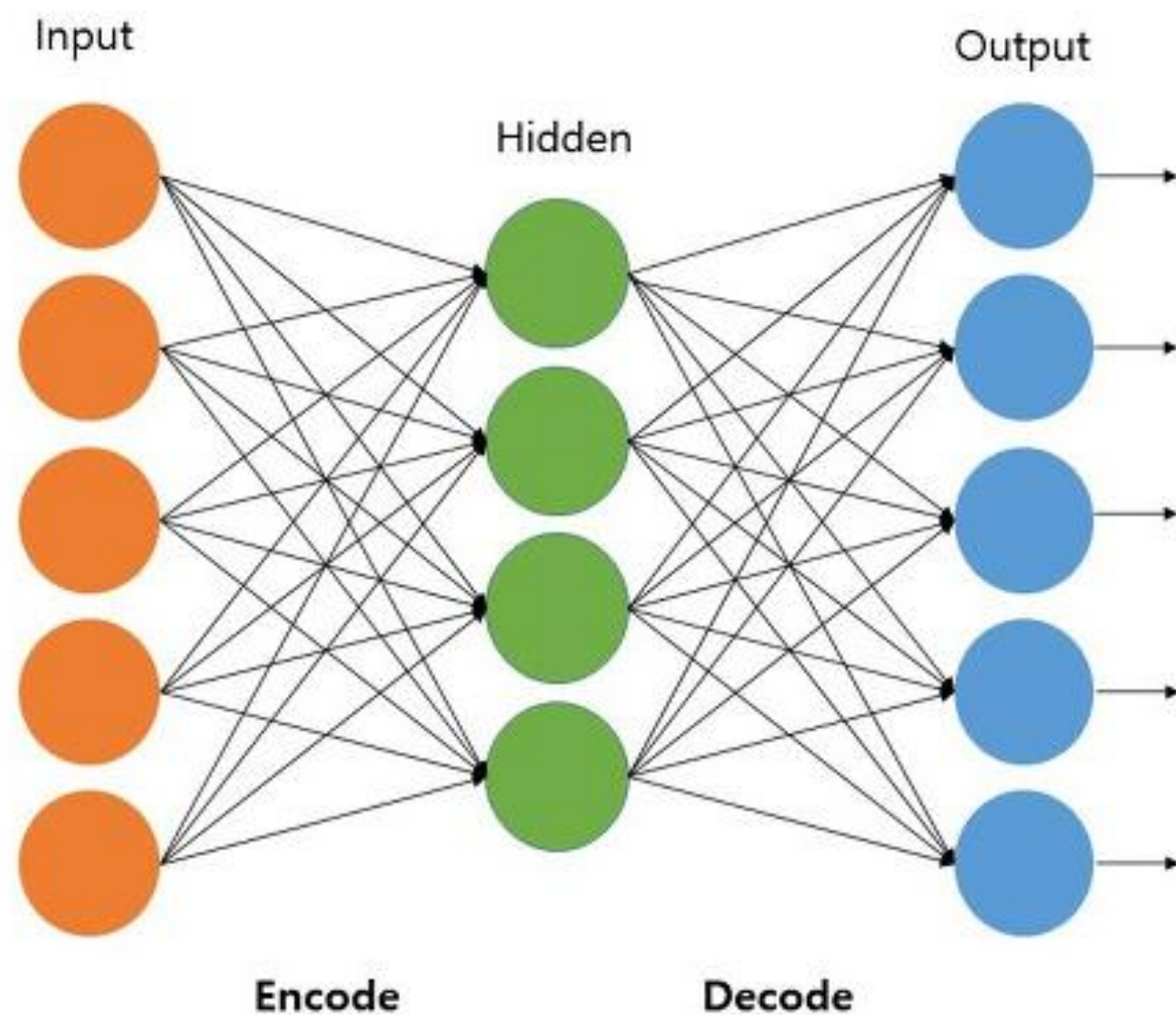


특징 벡터 차원 : 32



특징 벡터 차원 : 64

## 이상 탐지

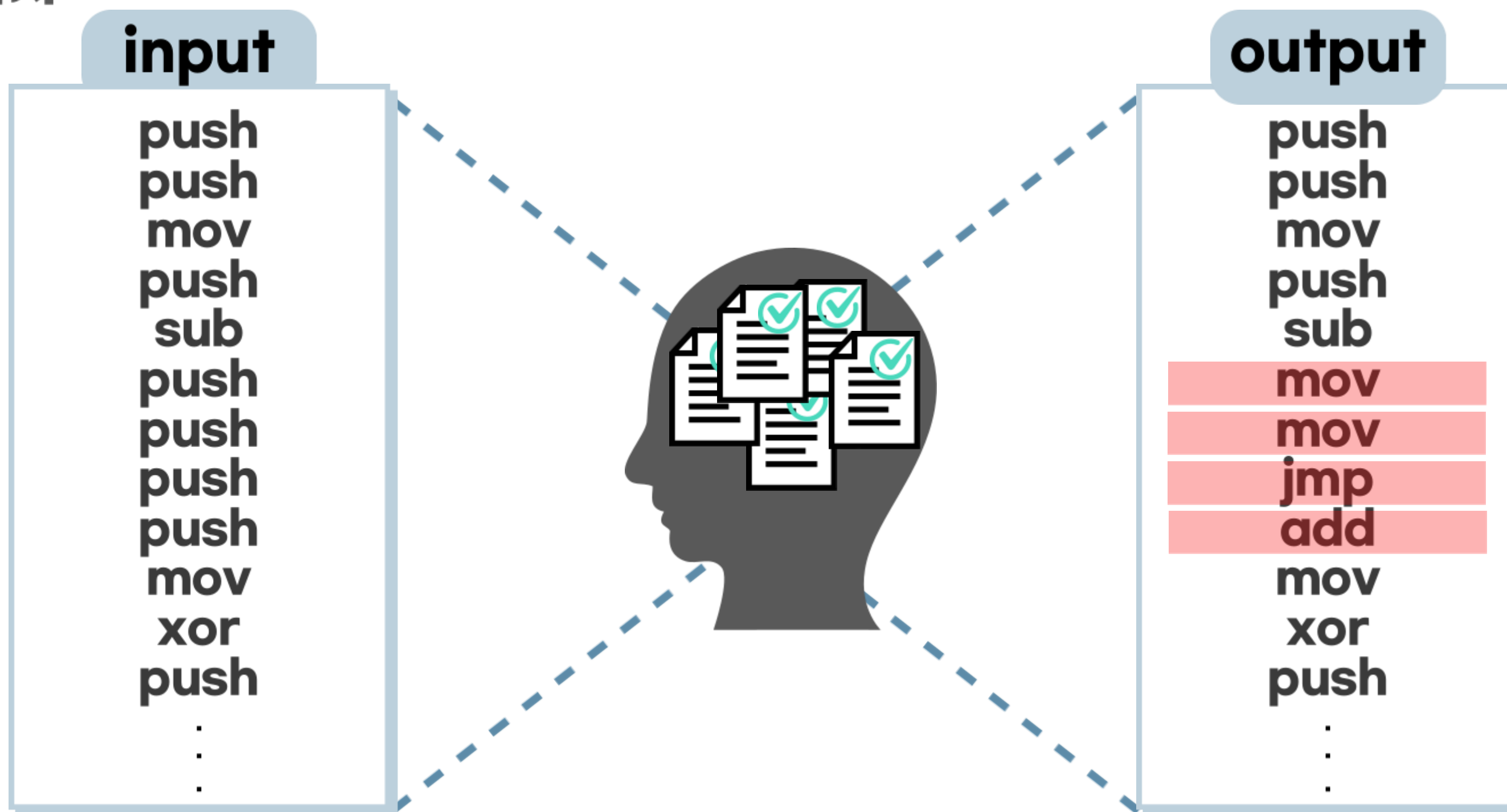


### 오토 인코더

대표적 비지도 학습법

입력 값과 출력 값을 같게 함

## 이상 탐지



## 이상 탐지

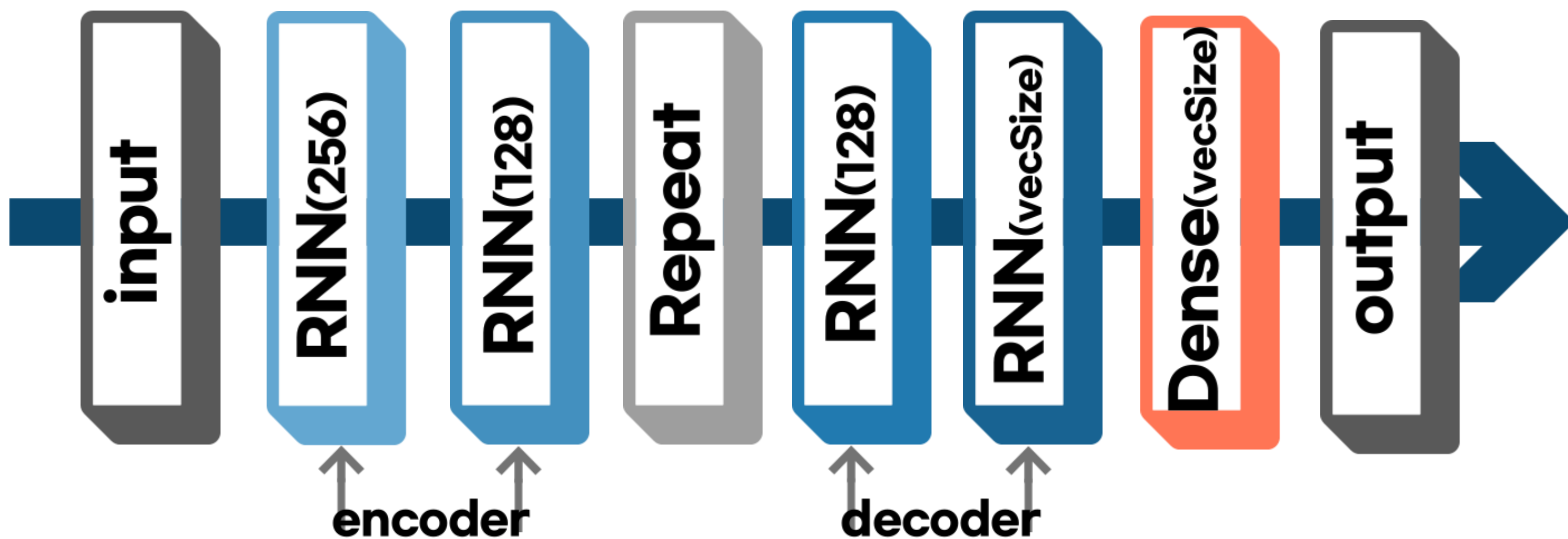
### | 실험 조건

- 01 임계점 : 0.1
- 02 벡터 크기 : 16 / 64
- 03 신경망 : GRU / LSTM
- 04 MAE(mean absolute error) 가 크면 이상탐지



이상 탐지

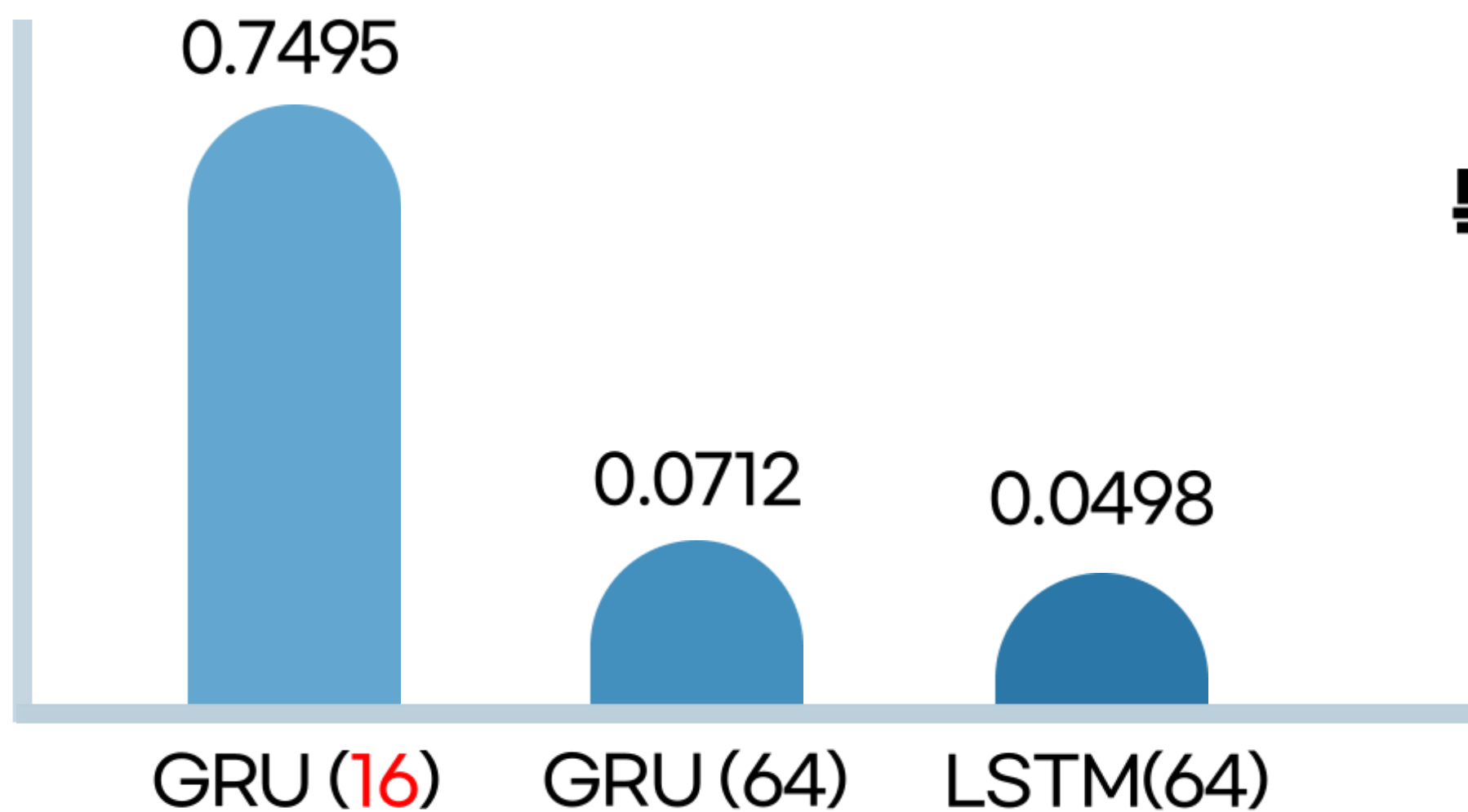
## | 신경망 구조





## 이상 탐지

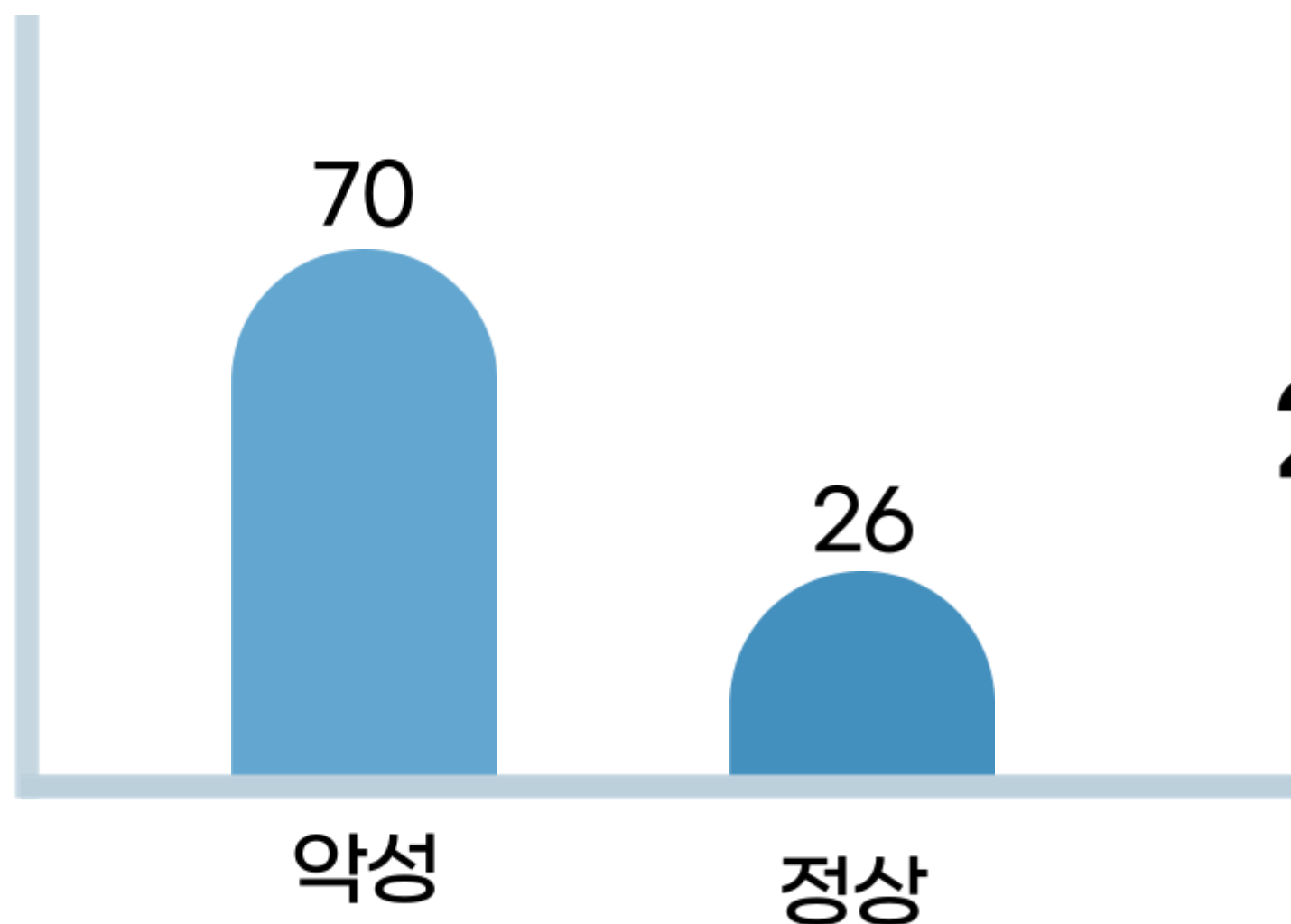
## | 실험 결과 - 손실값



특징벡터 차원  
최소 32차원

## 이상 탐지

### | 실험 결과 - 이상탐지



**악성이 정상보다  
2배 이상 이상탐지 됨**

## 검증 방법 제안 - 검색 엔진 활용

index	type	id, source
mal	_doc	<pre>_id : md5 source : push push mov push sub push push push push mov xor push l ea mov mov or push push call add mov mov push lea push lea push lea       ⋮ _id : md5 source : push push mov push sub push push push push mov xor push l ea mov mov or push push call add mov mov push lea push lea push lea</pre>
ben	_doc	<pre>_id : md5 source : push push mov push sub push push push push mov xor push l ea mov mov or push push call add mov mov push lea push lea push lea       ⋮ _id : md5 source : push push mov push sub push push push push mov xor push l ea mov mov or push push call add mov mov push lea push lea push lea</pre>

## 검증 방법 제안 - 검색 엔진 활용



RNN



jmp		00 00
mov		00 00
mov		00 00
call		00 00
xor		00 00
mov		00 00

이상탐지



파일 검색



정상 XX%



악성 XX%

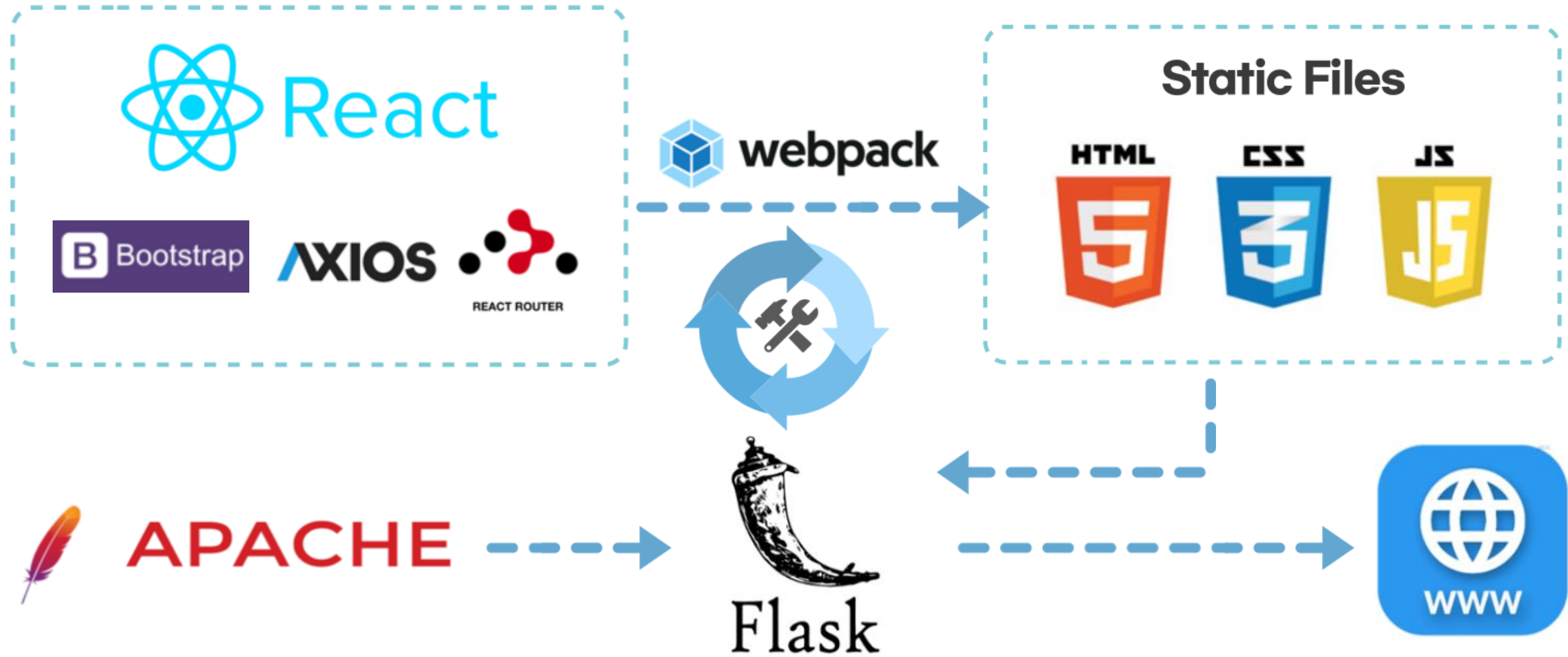
결과 내 악성 비율

악성 파일의 비율이 한계 값보다 클 경우



모델 판정 신뢰

## 웹 구현



## 웹 구현 - 업로드

Asi

서비스 소개

분석 요청

분석 요청하기

드래그로 업로드


파일 업로드

제출

## 드래그 & 드롭 방식



## 웹 구현 - 업로드 완료



[서비스 소개](#)[분석 요청](#)

분석 요청하기

00bd375cee6c303b6c351a2ae5ee262d.txt

00ca16afb348ebadc82975e47d88e274.txt

00d2c06a552f782c1f16acf77db765a5.txt

00d7ab9a8e5c9a84cfca19ad9e583e6f.txt

00d9dada816ad75caf87f9e09de2a521.txt


00d8914ba4c475e568a292afbd7b35d1.txt

00d150d750f70b000d50e075d70f447.txt

파일 업로드

제출

## 웹 구현 - 프로젝트 개요 페이지




[서비스 소개](#)
[분석 요청](#)

[View on GitHub](#)

### capstone-2020-5

capstone-2020-5 created by GitHub Classroom





a security insight

전문가를 위한 악성코드 분석 보조도구

I. 프로젝트 소개

New malware

Last update: March 12, 2020

Copyright © AV-TEST GmbH, www.av-test.org



01

프로젝트 목표

02

진행 상황

03

계획 및 제한요소

## 계획

## | 구현 계획

모델



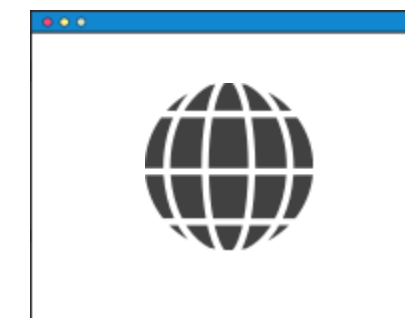
단어 임베딩 실험  
신경망 실험

검증



검색엔진 실험  
악성코드 보고서 분석

웹



분석 결과 페이지  
결과 다운로드 페이지

## 제한 요소



모델 학습 >>> 고사양 PC 요구

학교 제공 GPU 서버 >>> 현실적 사용 불가

## 월별 구현 계획

항목	세부내용	1월	2월	3월	4월	5월	6월
요구사항분석	요구 분석	☑					
	SRS 작성	☑					
관련분야연구	딥러닝 기술 연구		☑	☑			
	관련 논문 동향조사		☑	☑			
설계	시스템 설계				↻	↻	
구현	코딩 및 모듈 테스트				↻	↻	
테스트	시스템 테스트						☑

## 팀원 별 역할 분담



**손현기**

크롤러 & 파서 개발  
신경망 구현 및 튜닝  
웹 백엔드  
ELK 구축



**김주환**

논문 동향조사  
제안서 및 보고서 작성  
신경망 구현 및 튜닝



**김호준**

자료 조사  
문서작업 보조  
웹 프론트 개발



**오예린**

디자인  
웹 UI/UX 기획  
ELK 구축



**이동운**

정상파일 크롤러 개발  
신경망 구현 및 튜닝



**Ruslan**

opcode 파서 개발  
웹 프론트 개발



감사합니다.