최종 캡스톤 보고서

‘잡아라!’

프로젝트 명	잡아라!
팀 명	잡았조
팀원	
키워드	#몰래카메라 탐지 #좌표 측량 및 위치 추적 #패킷 캡쳐 및 분석
깃허브 주소	https://github.com/orgs/2025-Capstone-2/repositories

목차

프로젝트 개요

몰래카메라에 사용되는 카메라는 크게 3 가지로 분류할 수 있습니다. 내장 메모리를 가지고 있어 설치 후 회수가 필요한 Offline 카메라와 자체적으로 인터넷을 가져 서버나 클라우드에 영상을 저장하는 IP 카메라, 그리고 라디오 전파를 이용해서 데이터를 전송하던 RF(radio Frequency) 카메라가 존재합니다. 우리는 이 중에서 무선 네트워크를 이용하는 IP 카메라를 탐지하고자 합니다.

'잡아라'는 무선 네트워크 트래픽을 분석하여 의심스러운 기기를 찾고 해당 기기의 정확한 위치를 추적하는 어플리케이션입니다. 이 앱은 주변의 모든 네트워크 흐름을 감지한 후, 특정 기기가 보내는 맥 주소나 데이터 길이 등의 고유한 정보를 분석합니다. 특히 실시간 영상 스트리밍에서 발견되는 많은 양의 데이터 흐름을 집중적으로 탐색하여 정상적인 네트워크와 다른 이상 징후를 식별합니다. 또한 신호 세기를 거리로 역산하여 몰래카메라의 위치를 찾아냅니다. 이러한 방식으로 사용자는 일상적인 환경에서도 몰래카메라와 같은 불법 촬영 장비의 존재를 빠르고 효과적으로 감지할 수 있을 것으로 기대합니다.

모티베이션 및 솔루션

몰래카메라 범죄 증가

과거부터 지속적으로 몰래카메라 범죄는 심각한 사생활 침해 및 성범죄 사안으로 다루어졌습니다. 현재는 인터넷의 발달로 위장 카메라나 소형 카메라를 구하는 것이 수월해졌고, 과학 기술의 발달로 IP 카메라가 대두되면서 카메라를 회수할 필요 없이 실시간으로 서버 또는 휴대폰에 바로 영상을 보내는 몰래카메라가 개발되어 오고 있습니다.

여름 휴가철 증가하는 몰래카메라 범죄

충청일보 | 입력 2024.08.07 15:46 | 댓글 0

[생활안전이야기] 동중영 정치학박사·한국경비협회 중앙회장

지난 7월 30일 경찰청 통계에 따르면 지난 2023년 한 해 동안 카메라 등을 이용한 불법 촬영 발생 검거 건수는 6626건으로, 그중 7~8월이 1297건으로 약 20%를 차지한다고 밝혔다. 2020년에 개정된 '성폭력 처벌 등에 관한 특례법' 제14조에서는 신체를 촬영한 자는 7년 이하의 징역이나 5000만 원 이하의 벌금에 처하는 것을 명시하고 있으며, 유포하거나 소지한 자를 처벌하는 등 그 처벌 수위는 점차 강해지고 있다. 그런데도 여전히 매년 5000건 이상의 카메라 등 이용촬영 범죄가 일어나고 있다.

몰래카메라 범죄가 가장 자주 발생하는 장소로는 대기실, 그다음으로는 노상, 지하철, 아파트를 포함한 주택, 버스 등 교통수단에서도 범죄가 발생한다. 최근에는 숙박 공유 플랫폼의 서비스를 이용할 때도 이러한 피해가 다수 발생하여 논란이 되고 있다. 이와 같은 문제가 계속 발생하면 즐거운 여름 휴가철이 불안에 가득해지게 된다.

<그림 1> 기사 캡처[1]

법무부는 불법 촬영 범죄 증가 이유로 디지털 기기 보급의 일반화를 꼽았으며 특히 소형 디지털 기기에 대한 접근성과 무선 접속에 대한 낮은 난이도가 이를 더욱 용이하게 만들었다고 보고 있습니다. 충청일보 기사에 따르면 매년 5,000 건 이상의 불법 촬영 범죄가

[1] 충청일보, "여름 휴가철 증가하는 몰래카메라 범죄", 2024.08.07 https://www.ccdailynews.com/news/articleView.html?idxno=2290009

발생하고 있으며, 단추나 시계와 같이 일상적인 물건의 형태로 판매되어 육안으로 구분하기 어렵다고 합니다.

부정확한 해결책

고조되는 사람들의 불안감에 맞추어 여러 앱스토어에 핸드폰으로 몰래카메라를 탐지하는 앱들이 다수 출시되었습니다. 예를 들어 에스프레스토에서 개발한 '릴리의 지도'는 사진이나 전자파를 분석해 몰래카메라 탐지를 수행합니다. 또한 학교나 공공기관 차원에서는 화장실 입구에 필름지를 두고 수상한 곳을 직접 확인하도록 권유합니다.

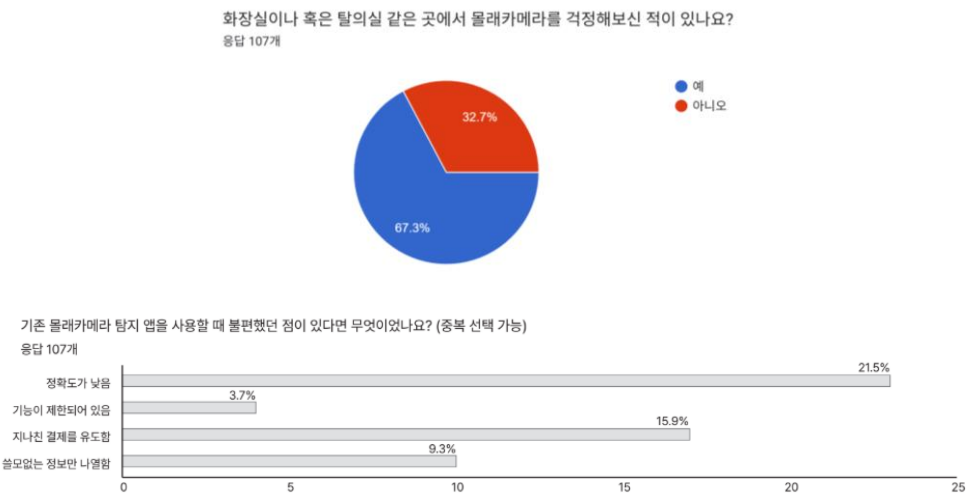
그러나 이러한 방법을 사용하는 앱은 원리가 불분명하고, 정확도가 매우 낮으며, 사실상 가짜인 앱이 대다수라는 문제가 있습니다. 그리고 사용자가 직접 필름지를 사용해 탐지하는 방법은 사용자의 역량에 따라 정확도가 결정되어 육안으로 탐지하기 어렵거나 예상할 수 없는 위치의 소형 카메라는 발견하기 어렵다는 단점이 있습니다.

정확한 해결책에 접근이 어려움

기존의 몰래카메라 탐지 방식은 주로 전파 탐지기나 적외선 탐지기를 활용하는 물리적 방법에 의존하고 있습니다. 하지만 이러한 방식은 몰래카메라의 신호 주파수를 변경하거나 적외선 반사 기능을 차단하는 등의 기술적 우회 방법이 등장하면서 한계를 보입니다. 또한 이러한 문제를 해결한 기기들은 전문 장비로 제작되어 가격이 높고 부피가 크기 때문에 일반인들이 접근하기 쉽지 않습니다.

설문조사

저희는 무작위 107 명을 대상으로 설문조사를 진행하였습니다. 그 결과 전체 인원 중 약 ⅔ 이상이 화장실 혹은 탈의실 같은 장소에서 몰래카메라를 걱정해 본 경험이 있다고 답하였으며, 기존 몰래카메라 탐지 앱을 사용할 때 불편했던 점으로 정확도가 낮다는 것과 지나친 결제를 요구함을 문제로 꼽았습니다. 이러한 설문조사 결과에 따라 저희는 개인이 접근 가능한 탐지 방법에는 명확한 한계가 있음을 확인할 수 있었습니다.



<그림 2> 설문조사 결과

솔루션

이러한 문제점들을 해결하기 위해 '잡아라'에서는 몰래카메라 탐지를 더욱 정밀하고 효과적으로 수행할 수 있는 소프트웨어를 개발하여 개인과 공공시설의 보안을 강화하고 불안을 줄이는 솔루션을 제공하고자 합니다.

'잡아라'는 기존의 물리적 및 상용화된 탐지 방식이 가진 한계를 극복하여 몰래카메라의 존재 여부뿐만 아니라 정확한 위치까지 파악할 수 있도록 합니다. 이를 통해 사용자는 단순한 의심 단계에서 벗어나 직접적으로 불법 촬영 장비를 제거할 수 있게 되어 지속적인 감시와 대응이 가능해집니다. 최종적으로 저렴한 LAN 카드를 통해 모든 사람이 쉽게 접근할 수 있도록 개발하여 전문적인 보안 지식이 없는 일반인도 일상생활에서 사생활을 보호할 수 있도록 하는 것을 목표로 하고 있습니다.

차별점

‘잡아라’는 기존 서비스들과 비교했을 때 다음과 같은 차별점들을 가집니다.

1. 신뢰성

전자기기 이외의 기존 탐지 방법들은 사용자의 관찰 능력에 의존하거나 검색되는 모든 와이파이를 보여주어 몰래카메라의 유무를 알기 어려웠습니다. 그러나 ‘잡아라’는 해당 기기가 내보내는 데이터의 흐름을 확인하고 알려주기 때문에, 사용자의 판단 능력에 좌우되지 않아 신뢰성이 높습니다. 또한 거리 및 와이파이 세기에 따른 필터링도 포함하여 사용자에게 혼선을 주지 않습니다.

2. 접근성

‘잡아라’는 휴대폰과 부속 기기 하나 만으로 몰래카메라의 존재 여부를 알 수 있고, 삼변측량으로 스스로 몰래카메라 기계를 찾을 수 있게 하였습니다. 준비해야 하는 기기들은 쿠팡이나 아마존과 같은 인터넷을 통해서 비교적 저렴한 가격으로 어렵지 않게 구매할 수 있기 때문에 사용자는 복잡한 절차나 과한 소비 없이도 몰래카메라를 찾아낼 수 있습니다.

3. 가격

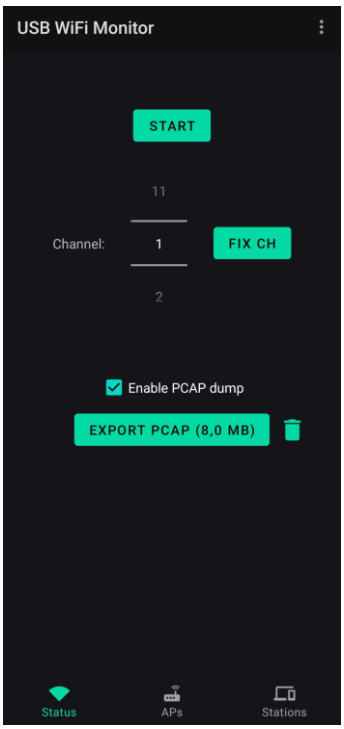
신뢰성과 정확도를 모두 갖춘 기존 탐지기(전자기기)는 작은 것은 10 만 원에서 크고 강력한 것은 75 만 원까지의 비용을 필요로 합니다. 그러나 ‘잡아라’는 1 만원 대에서 저렴한 기기를 구매하는 것 만으로 효과를 볼 수 있기 때문에 과도한 비용을 들이지 않고 몰래카메라를 탐지할 수 있습니다. 이를 통해 예산 절감 효과를 기대하고 있습니다.

	비용	정확성 (카메라 위치 찾기)	신뢰성 (카메라 유무 판단)
전자기기	상	o	o
몰카감지기(앱)	하	x	x
잡아라	하	o	o

개발 내용

네트워크 패킷 캡처

본 프로젝트는 네트워크 패킷을 캡처하기 위해 모니터 모드를 사용하며, 이를 루팅 없이도 가능하게 해주는 USB WIFI Monitor 라는 앱을 같이 사용합니다. 이는 모니터 모드로의 전환에서 필요했던 휴대폰의 루팅 과정은 핸드폰을 고장 낼 우려가 있어 해당 방법을 채택하게 되었습니다. 이 앱을 사용하기 위해서는 AR9271 칩셋 기반 LAN 카드와 이를 휴대폰에 연결할 OTG 케이블이 필요합니다.



<그림 3> USB WIFI Monitor

USB WIFI Monitor 앱은 루팅 없이도 모니터 모드의 실행을 통해 네트워크 패킷을 캡쳐 할 수 있도록 해줍니다. 초기에는 이러한 앱 없이도 자체적으로 캡처가 가능한 기능도 구현하려고 했지만, 휴대폰 내부 혹은 외부 LAN 카드를 이용하는데 있어 루팅이 필수적이었습니다. 해당 앱은 루팅 없이 외부 LAN 카드를 우회해서 사용하고 있었고, 저희는 앱의 개발자인 Emanuele Faranda 씨에게 연락을 드려 조언을 받고자 하였습니다.

Emanuele Faranda 씨는 코드를 공개할 생각은 없지만 대신 다른 앱을 통한 제어가 가능하도록 API 를 개발하여 제공해 주셨습니다. API 를 통해서 '잡아라'는 캡처 시작/종료

제어가 가능해졌고, 이렇게 캡처한 pcap 파일을 백엔드로 보내어 분류와 탐지를 실행합니다.

거리 최적화

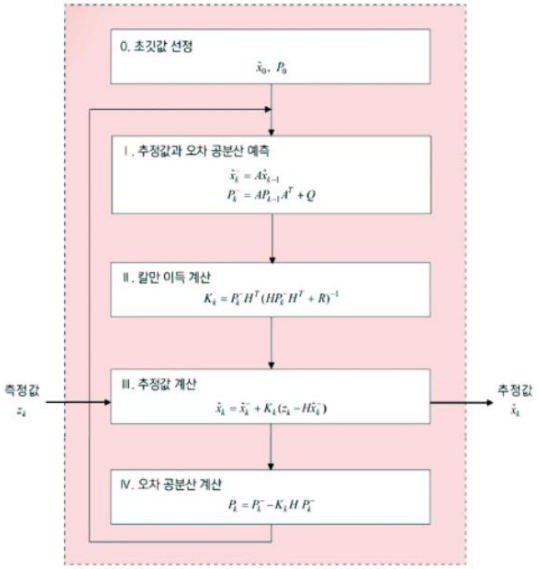
와이파이 신호 세기는 주변 환경 및 다양한 기기로 인해 간섭이 발생하며, 이 때문에 RSSI 값에 노이즈가 끼어 이상적인 일정한 값을 얻을 수 없습니다. 따라서 이 문제를 해결하기 위해 저희는 다양한 신호 세기 보정 방법을 조사하였고, RSSI 기반 위치 탐지를 연구한 논문[2][3]에서 활용된 칼만 필터를 채택하게 되었습니다.

칼만 필터

칼만 필터(Kalman Filter)는 잡음이 있는 측정값을 기반으로 선형 역학계의 상태를 추정하는 재귀 필터 알고리즘으로, 과거에 수행한 측정값을 바탕으로 현재 상태 변수의 결합분포를 추정합니다. 칼만 필터는 물체의 측정값에 확률적인 오차가 포함되고, 또한 물체의 특정 시점에서의 상태가 이전 시점의 상태와 선형적인 관계가 있는 경우 적용이 가능합니다.

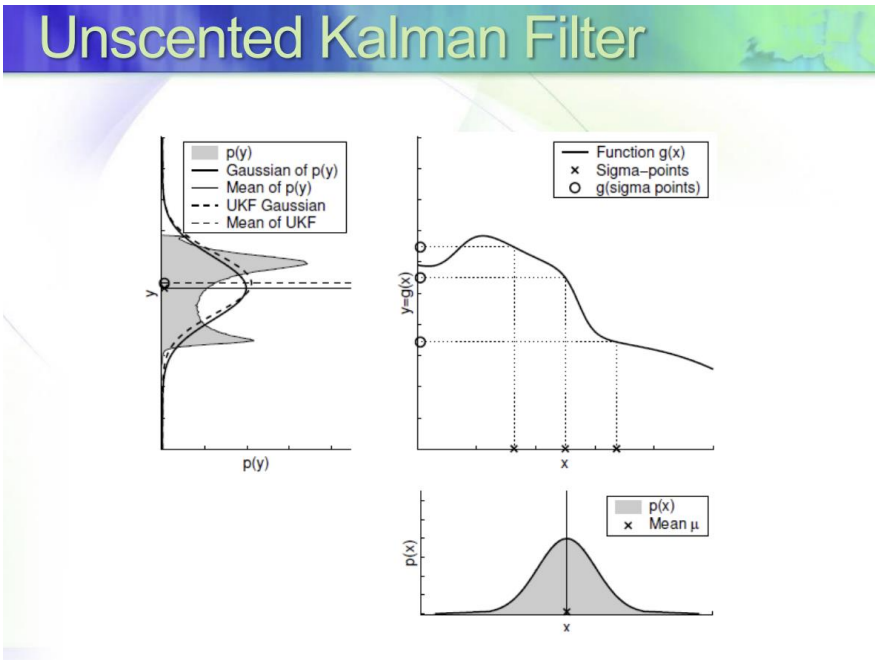
알고리즘은 예측과 업데이트라는 두 단계로 이루어집니다. 예측 단계에서는 이전 시간에 추정된 상태에 대해, 그 상태에서 사용자 입력을 가했을 때 예상되는 측정값을 계산합니다. 현재 상태 변수의 값이 실제로 측정된 이후, 업데이트 단계에서는 이전에 추정한 상태 변수를 기반으로 예측한 측정치와 실제 측정치의 차이를 반영해 현재의 상태 변수를 업데이트합니다.

[2] Syifaul Fuada, Trio Adiono, and Prasetiyo, “Improvement of RSSI-based Distance Localization using Unscented Kalman Filter (UKF) Algorithm for Wi-Fi Tracking Application”, 2019 iJIM.
[3] Bo Yang, Xinchun Jia, and Fuwen Yang, “Variational Bayesian Adaptive Unscented Kalman Filter for RSSI-based Indoor Localization”, 2021 Journal of Control.



<그림 4> KF 알고리즘

다만 전술하였듯이 일반 칼만 필터는 대상의 전후 상태가 선형적인 관계가 있는 경우에만 적용이 가능하기 때문에, 저희는 비선형 관계에도 적용이 가능한 무향 칼만 필터(Unscented Kalman Filter)를 고려하였습니다.



<그림 5> 무향 칼만 필터[4]

[4] Mohamed Attia Aref, Kalman Filtering (from basics to unscented kalman filter), https://www.slideshare.net/SCU_ECE_Staff/kalman-filtering-52057995

무향 칼만 필터는 비선형 시스템이 함수 f(x)를 따른다고 가정할 때, 가우시안 분포를 통해 이 f(x)를 잘 나타낸다고 판단되는 sigma points 를 찾습니다. 이 sigma points 는 가우시안 분포의 평균에서 멀어질수록 가중치가 줄어드는 방식으로 새로운 업데이트에 영향을 끼치게 되고, 함수 f(x)에 직접 넣는 과정을 가집니다. 계산량이 많다는 단점이 있지만 안드로이드 기기로 충분한 성능이 나오며, 정확도를 가장 높아 해당 필터를 채택하였습니다.

머신러닝

데이터 수집

학습을 하기 위해서는 적절한 데이터가 필요하며, 다양한 기기에서도 의도대로 작동하도록 두 종류의 몰래카메라(무선 카메라 모듈)를 구매하였습니다.

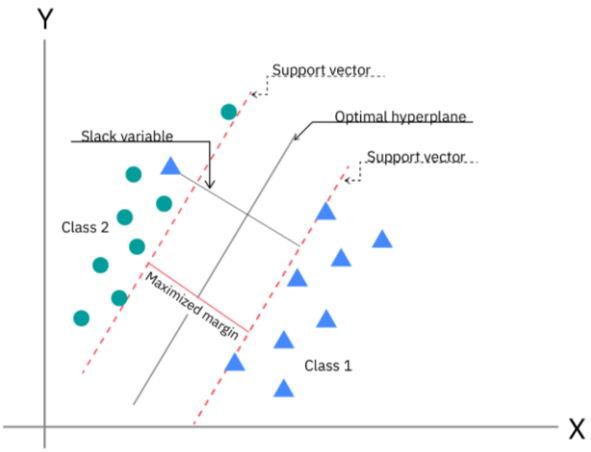
<그림 6> 구매한 몰래카메라 기기

카메라와 일반 무선 신호와의 구별을 위해 자체 제작한 테스트베드에서 각각의 카메라를 구동시킨 뒤 1 분 길이의 패킷 파일 400 개씩 총 800 개를 수집하였으며, 그 외에도 200 개의 카메라가 없는 일반 환경의 패킷 파일을 포함하여 총 1000 개의 PCAP 파일을 수집하였습니다. 이 중 80%인 800 개의 파일을 학습 용도로, 20%인 200 개의 파일을 테스트 용도로 사용하였으며 파일 비율은 4:4:2 로 일정하게 설정하였습니다.

알고리즘 모델

데이터 학습을 위해 LSTM, k-NN, XGBoost 등의 다양한 알고리즘 모델을 고려했지만, 최종적으로는 SVM(Support Vector Machine)을 선택하였습니다.

SVM 은 N 차원 공간에서 각 클래스 간의 거리를 최대화하는 최적의 선 또는 초평면을 찾아 데이터를 분류하는 지도형 머신 러닝 알고리즘으로, 서로 반대되는 두 클래스의 가장 가까운 데이터 포인트 사이의 마진을 최대화하는 최적의 초평면을 찾아 두 클래스를 구분합니다. 클래스를 구분하기 위해 여러 초평면을 찾을 수 있으므로 포인트 사이의 마진을 최대화하면 알고리즘이 클래스 간 최적의 결정 경계를 찾을 수 있어, 이를 통해 새로운 데이터에도 잘 일반화하여 정확한 분류 예측을 할 수 있습니다.



<그림 7> SVM 시각화 그래프[5]

SVM 을 선택한 이유는 다른 ML 모델들에 비해 비해서 비교적 적은 샘플만으로 경쟁력 있는 분류 정확도를 낼 수 있고, 학습 데이터가 모델 파라미터로 남지 않으므로 타 모델보다 훨씬 가볍고 메모리 효율이 높았기 때문입니다. 또 실시간 성능이 중요하기 때문에 이에 적합하다고 판단되어 사용하게 되었습니다. 유사한 ML 모델인 XGBoost 과 테스트하여 비교했을 때 SVM 은 더 적은 파라미터로도 충분한 정확도를 보여준 반면, XGBoost 는 더 많은 파라미터를 요구하기 때문인지 과적합을 보여주어 이번 프로젝트에 적합하지 않다고 판단하여 SVM 을 채택하게 되었습니다.

[5] IBM, “서포트 벡터 머신(SVM)이란 무엇인가요?”, https://www.ibm.com/kr-ko/think/topics/support-vector-machine

분류 기준

몰래카메라의 패킷을 포착하기 위해 우선 전체 패킷들을 스트림 단위로 분할하였습니다. 하나의 스트림은 Source 와 Destination 의 MAC 주소로 결정되며, 이 스트림에 카메라가 포함되어 있는지 아닌지를 구별하는 것이 목표입니다. 이는 패킷을 일일이 처리하는 것은 매우 오랜 시간이 걸리고, 윈도우 단위로 처리하는 것도 한 번의 추출 과정에서 많은 데이터를 처리하는 것을 요구하기 때문에 선택한 추출 방법입니다.

몰래카메라 스트림과 일반 스트림의 차이를 밝혀내기 위해 여러 논문들에서 말한 '몰래카메라 스트림은 대용량의 패킷을 전송할 것이다' 라는 내용을 확인하기 위해 스트림에서 다음과 같은 데이터를 추출하도록 하였습니다.

- src_mac(송신자 MAC 주소), dest_mac(수신자 MAC 주소), total_packets(스트림 내 전체 패킷 수), mean_interval(패킷 간 평균 전송 간격), std_interval(패킷 간 전송 간격 표준편차), mean_length(패킷 길이의 평균값), std_length(패킷 길이 표준편차), inbound_bytes(스트림으로 들어온 총 바이트 수), duration(스트림 시작부터 종료까지 지속 시간), packet_rate(단위 시간당 전송된 패킷 수), byte_rate(단위 시간당 전송된 바이트 수), min_interval(패킷 간 최소 전송 간격), max_interval(패킷 간 최대 전송 간격), median_interval(패킷 간 전송 간격 중앙값), burstiness(간격 버스트성), interval_entropy(패킷 간 전송 간격 분포의 엔트로피), interval_skew(전송 간격 분포의 왜도), interval_kurt(전송 간격 분포의 첨도), cv_interval(전송 간격 변동 계수), length_min(패킷 길이 최솟값), length_max(패킷 길이 최댓값), length_median(패킷 길이 중앙값), length_percentile25(패킷 길이 25 백분위값), length_percentile50(패킷 길이 50 백분위값), length_percentile75(패킷 길이 75 백분위값), length_entropy(패킷 길이 분포의 엔트로피), length_skew(패킷 길이 분포 왜도), length_kurt(패킷 길이 분포 첨도), unique_packet_sizes(서로 다른 패킷 길이 개수), num_large_pkts(≥1000 바이트 패킷 개수), small_packet_ratio(<100 바이트 패킷 비율), large_packet_ratio(>800 바이트 패킷 비율), cv_length(패킷 길이 변동 계수)

그리고 두 카메라의 스트림과 일반 스트림에 대한 정보를 비교해본 결과, 다음의 차별점을 얻어 이를 분류 기준으로 삼게 되었습니다.

- 패킷의 크기가 매우 크거나 작다.(total_packets)

- 일반적으로 100 이상의 크기를 가진 패킷은 Heavy-Hitter 라고 불리며, 보통은 특수 클래스로 분류됩니다. 1 의 크기를 가진 패킷도 IoT 비컨의 특성으로서 별도의 후보군으로 두어집니다.[6][7]

- 평균 간격이 짧다.(mean_interval)
  - 비디오 및 카메라 스트림은 지속적으로 데이터를 밀어 보내기 때문에 평균 간격이 짧다는 특징을 가지고 있습니다. 유튜브와 같은 영상 서비스도 동일한 성질을 가집니다.[8][9]

- 패킷 분포가 집중되어 있다.(length_entropy)
  - 패킷 길이가 고정되어 있으면 길이 히스토그램의 엔트로피가 매우 작아집니다. 이는 패킷이 변동하지 않으며, 길이 분포가 한 곳에 집중됨을 의미합니다.[10][11]
  - 단발성 패킷은 패킷 분포가 0 이 되므로 이를 배제하는 것을 염두에 두어야 합니다.

[6] Towards threshold-agnostic heavy-hitter classification, Adrian Pekar, Alejandra Duque-Torres, Winston K.G. Seah, Oscar M. Caicedo Rendon

[7] Notes on the per-flow packet count flow classifier, Mika Ilvesm¨aki, Jouni Karvo

[8] Network Characteristics of Video Streaming Traffic, Ashwin Rao, Yeon-sup Lim, Amherst

[9] IoT Device Identification Using Directional Packet Length Sequences and 1D-CNN, Xiangyu Liu, Yi Han, Yanhui Du

[10] An Entropy-Based Network Anomaly Detection Method, Przemysław Bereziński,Bartosz Jasiul, Marcin Szpyrka

[11] Device Identification Method for Internet of Things Based on Spatial-Temporal Feature Residuals, Shi Dong, Longhui Shu, Qinyu Xia

학습 및 테스트

학습은 모든 PCAP 데이터 파일을 .csv 형식으로 추출함과 동시에 위의 분류 기준에 따라 라벨링을 한 뒤 SVM 을 통해 진행됩니다. 위의 세 가지 조건을 모두 만족해야 해당 스트림은 1 로 라벨링 되고, 그렇지 않으면 0 으로 라벨링 됩니다.

초기에는 육안으로 공통적인 범위를 찾아서 값이 거의 정확하게 일치하거나 좁은 범위에 해당해야 1 로 라벨링 되도록 하였지만, 이 방법으로 시도하였더니 기본적인 정확도가 낮고 과적합이 자주 발생하였습니다. 그래서 이를 방지하고 정확도를 높이기 위해 일부 구간은 더 엄격하게, 일부 구간은 더 느슨하게 범위를 설정하는 등 파라미터의 튜닝을 진행하였습니다.

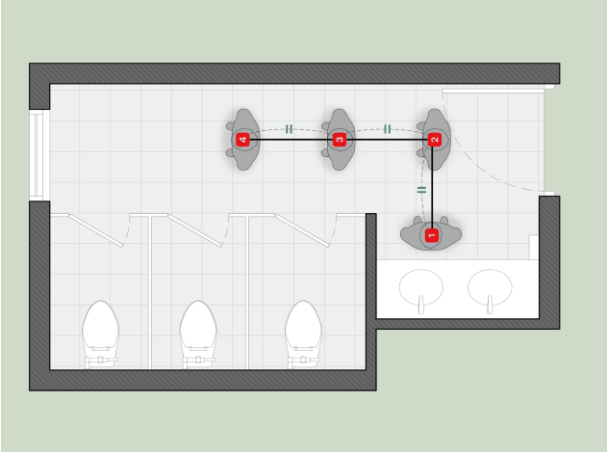
```
Cached cache\train: X shape (3436, 9), y shape (3436,)
Cached cache\test: X shape (912, 9), y shape (912,)
Train shape: (3436, 9) (3436,)
Test  shape: (912, 9) (912,)
5-fold CV ROC AUC scores: [0.94872699 0.9691934  0.93683838 0.97150086 0.92460277]
Mean CV ROC AUC: 0.9501724796928105

=== Test Classification Report ===
              precision    recall  f1-score   support

           0     0.9878    0.8858    0.9340       639
           1     0.7847    0.9744    0.8693       273

    accuracy                         0.9123       912
   macro avg     0.8862    0.9301    0.9016       912
weighted avg     0.9270    0.9123    0.9146       912

Test ROC AUC: 0.9371327681186836
Best params: {'svc__C': 100, 'svc__gamma': 1}
Best CV AUC: 0.9998687684415415
```

<그림 8> 테스트 결과

이렇게 다양한 파라미터로 실험한 결과 패킷의 크기가 100 이상이거나 1, 평균 간격이 0.6 이하, 패킷 분포도가 1.2 미만인 조건에서 가장 정확도가 높게 나오는 것을 확인할 수 있었습니다. 이렇게 하여 약 90% 이상의 정확도(f1-score)와 98% 이상의 정밀도(precision), 78% 이상의 재현도(recall)를 획득하게 되어 실제 사용에 적합하다고 판단하게 되었습니다.

이렇게 학습을 통해 생성된 기계학습 모델은 .pkl 파일로 생성되며, 이는 백엔드에서 PCAP 파일을 입력 받고 해당 모델로 평가됨으로써 몰래카메라 스트림을 추적하게 됩니다.

삼변측량



<그림 9> 화장실 내에서의 측정위치 예시

'잡아라'는 수상한 기기의 정확한 위치 좌표를 얻기 위해 삼변측량을 이용합니다. 사진과 같이 4 지점에서 기기에 대해 RSSI 를 측정하여 거리 정보를 얻은 뒤, 해당 거리를 반지름으로 하는 구를 그려서 생기는 교점을 카메라 위치로 정의합니다. 얻어진 좌표들의 기준점(원점)은 사진에서 보이는 1 번 위치(최초 위치)를 기준으로 하였습니다.

그러나 삼변측량은 거리 정보가 알맞다는 전제하에 설계된 결정론적 알고리즘입니다. RSSI 는 현실에서 측정한 물리적인 신호 세기이기 때문에 노이즈가 무조건 존재하며, 거리 정보에 영향을 주어 공식의 해가 없는 경우가 발생할 수 있습니다. 따라서 최소자승법을 통해 오차값은 최소화하면서 근사해를 찾도록 개발하였습니다.

$$\min_{x,y,z} \sum_{i=1}^{n} \left(\sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2} - d_i \right)^2$$

최소자승법은 다음과 같습니다. 네 곳의 측정지들의 중앙값을 초깃값으로 정하고 해당 위치에서 측정지까지의 거리를 계산합니다. 그 뒤, 측정한 거리의 차이만큼 오차를 보정하는 과정을 거치며 여러 번 반복해서 거리 오차를 최소화하는 근삿값을 찾게 됩니다. 다음 방식을 통해 저희는 원점(1 번 측정 위치)을 기준으로 수상한 기기의 x, y, z 좌표를 계산하였습니다.

기타

Backend

'잡아라'는 Python 의 FastAPI 라이브러리를 사용하여 구현하였습니다. '잡아라'의 Backend 는 삼변측량 및 머신러닝 기능의 연결과 관리를 담당하고, Frontend 의 API 요청을 수행합니다. Backend 의 관리를 위하여 Render 에서 제공하는 클라우드 호스팅 서비스를 활용했습니다.

Frontend

'잡아라'은 Kotlin Jetpack 을 통해 앱 인터페이스를 제공합니다. 기존에 존재하는 다양한 앱 UI/UX 를 분석하여 기본적인 프레임을 디자인한 뒤, '잡아라'의 Design System 의 색상을 적용했습니다. 내부 개발에서는 MVVM 구조를 채택하여 구현하였으며, API 연동을 위해서 RetrofitManager 를 사용했습니다.

기능 명세

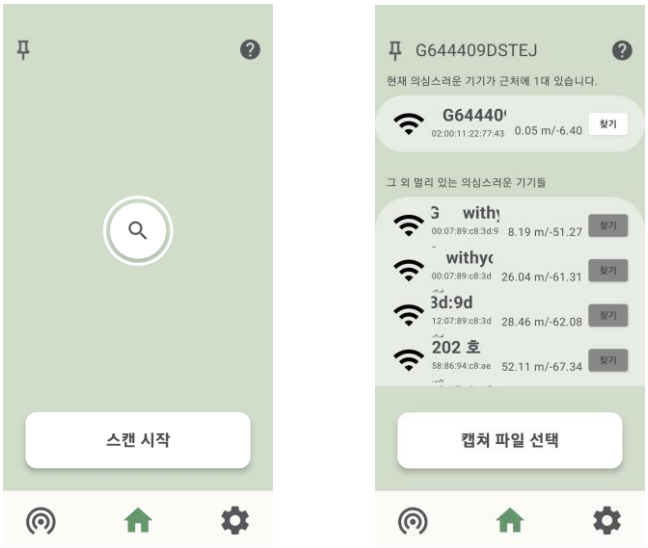
'잡아라'가 제공하는 기능을 사용 흐름에 맞춰 설명하겠습니다.

0. 준비물

네트워크 캡쳐를 위해 Google Play 에서 "USB WiFi Monitor" 1.2.0 버전을 다운로드 했습니다. 이 앱은 AR9271 칩셋으로 구동되는 LAN 카드를 모니터 모드로 전환 및 패킷 캡쳐를 지시할 수 있는 API 를 포함하고 있습니다. 따라서 AR9271 칩셋 LAN 카드가 필요한데 약 15000 원에 "Atheros AR9271" 모델을 구매하여 사용하였으며, 안정적인 전원 공급을 위해 OTG 케이블을 준비했습니다.

마지막으로 단위 거리, 신체 길이를 측정할 줄자도 준비했습니다.

1. 네트워크 스캔 과정



<그림 10> 메인 화면, AP 목록

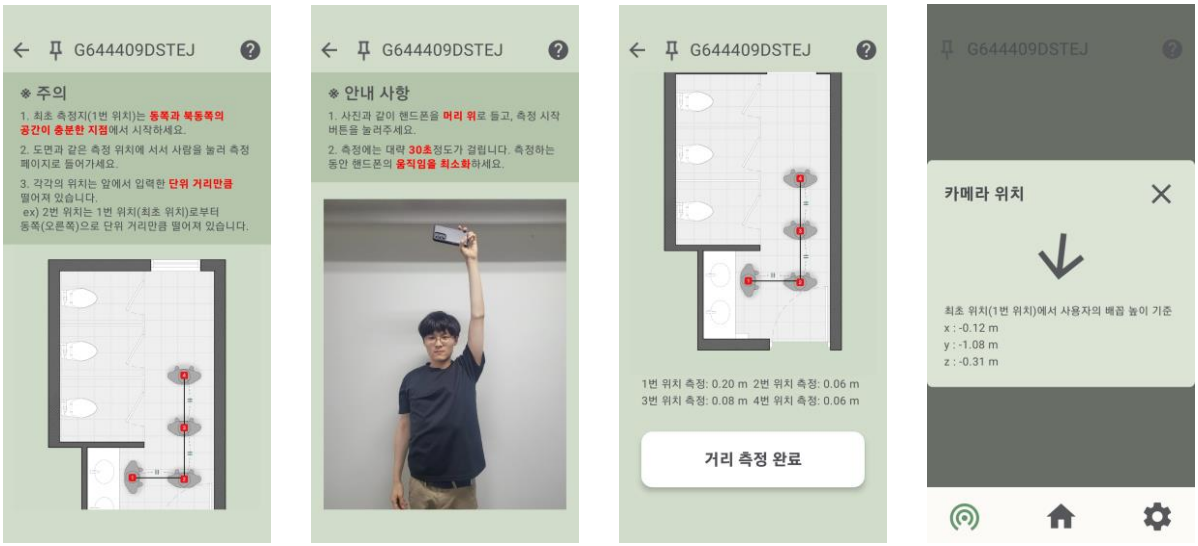
처음 앱에 들어가면 다음과 같은 메인 화면을 마주하게 됩니다. LAN 카드를 연결하고 스캔 시작 버튼을 누르면 주변 AP 목록을 볼 수 있고, 캡쳐 파일 선택을 통해 PCAP 파일을 앱에 전달하면 머신러닝으로 수상한 기기가 존재하는지 확인할 수 있습니다. 수상한 기기에 대해 찾기 버튼을 눌러 위치 찾는 과정으로 넘어갈 수 있습니다

.

2. 몰래카메라 위치 탐색 과정

<그림 11> 카메라 위치 찾기 화면

하단 왼쪽 탭에서 카메라 위치 찾기 버튼을 누르면 몰래카메라의 정확한 위치를 찾는 과정으로 넘어갈 수 있습니다. 처음 버튼을 누르면 측정에 필요한 길이 단위들을 사용자의 상황에 맞게 입력할 수 있습니다.

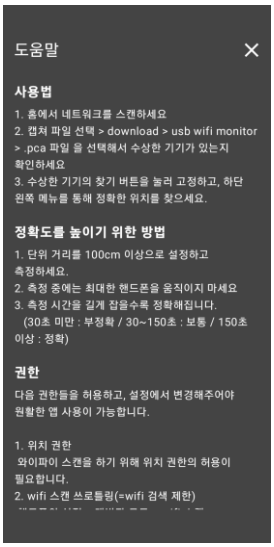


<그림 12> 거리 측정 가이드 및 최종 위치 화면

주의 사항에 맞추어 각 위치에서 측정할 수 있습니다. 측정 위치를 누르면 측정하는 방법을 안내하는 페이지로 넘어가고, 지시하는 자세에 맞추어 측정을 진행할 수 있습니다. 네

지점에서 모두 측정이 완료되면 거리 측정 완료 버튼을 통해 카메라의 정확한 위치를 얻게 됩니다. 위치는 1 번 위치를 기준으로 하여 알려줍니다. 사용자는 방향과 높이 값을 참고하여 카메라를 찾을 수 있습니다.

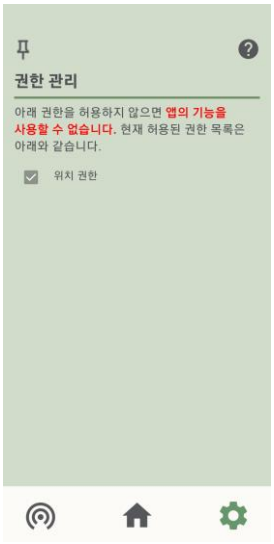
3. 도움말 페이지



<그림 13> 도움말 화면

사용자는 모든 페이지에서 우측 상단의 물음표 버튼을 통해 도움말을 확인할 수 있습니다. 앱의 사용법, 정확도를 높이는 방법, 필요한 권한 외에 앱의 동작 원리, 취지 등이 보이는 페이지입니다.

4. 설정 페이지



<그림 14> 설정 화면

사용자는 설정 페이지에서 앱에게 허용한 권한 목록을 확인할 수 있습니다. 현재는 와이파이 스캔을 위한 위치 권한만이 필요로 하는 것을 확인할 수 있습니다.

개선사항

패킷 파일 미사용

현재 '잡아라'는 패킷 캡처를 수행한 뒤, 이를 파일로 저장한 다음 분석을 하는 방식으로 탐지를 수행하고 있습니다. 이는 통신 보호 비밀법에 저촉될 우려가 있으며, 패킷의 암호화를 풀지 않아도 법적으로 문제가 될 수 있습니다. 이를 해결하기 위해서 패킷 캡처를 한 결과를 파일로 저장하지 않고 캡처를 한 스트림을 실시간으로 분석하거나 불필요한 정보는 마스킹 처리하도록 개선하고자 합니다.

사용자 지정 측정 시간

저희가 사용하고 있는 칼만 필터는 연구에 따르면 RSSI 노이즈를 안정화하는데 평균 150 초의 시간이 소요됩니다. 이는 사용자가 약 2 분 30 초 가량 팔을 들고 있어야 하므로 직접 측정하기에 무리가 있을 정도로 긴 시간이어서 사용성이 크게 저하된다고 판단하였습니다. 이를 보완하기 위해 저희는 사용자가 직접 측정 시간을 조정할 수 있도록 하여 사용성과 정확도 사이의 조정을 사용자가 할 수 있도록 설계하였습니다. 이를 통해 사용자는 짧은 시간 동안 측정하여 대략적인 위치를 파악하거나, 충분히 긴 시간 동안 측정하여 더 정확한 위치를 얻을 수 있게 됩니다.

머신러닝 학습 한계

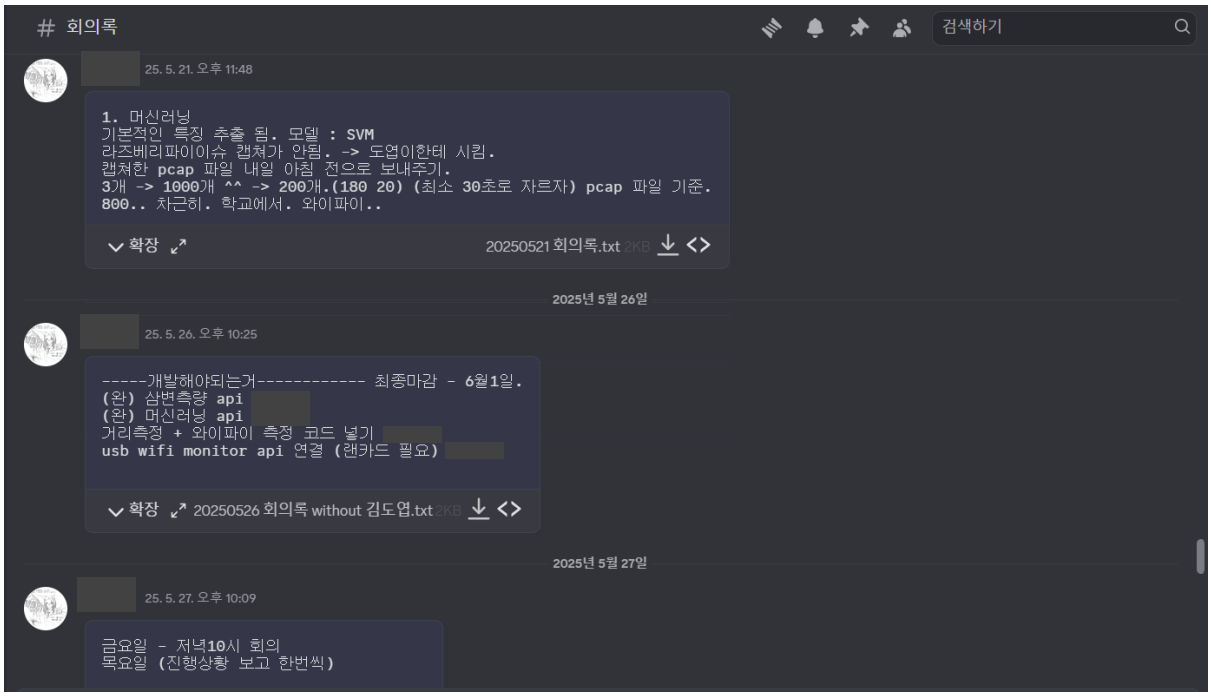
경제적 여건의 한계로 학습 및 테스트 용도의 IP 카메라를 2 대만 구입하여 프로젝트를 진행했습니다. 따라서 머신러닝 학습 자료가 카메라의 종류가 다양하지 않기 때문에 얻을 수 있는 데이터가 많지 않다는 아쉬움이 존재했습니다. 이로 인해 머신러닝에서 과적합이 존재하는 것으로 추정됩니다. 그러나 이는 시간과 예산이라는 현실적인 문제로 인해 불가피한 선택이었으며 이 문제가 해결된다면 충분히 해소 및 확장 가능한 부분이라고 생각됩니다.

측정 정확도

현재 대략 30 번의 시도 결과 중 60%가 오차 범위 30 초 이내의 결과를 보이고 있습니다. 원인 분석을 위해 각 케이스 별로 나누어 테스트를 진행해 본 결과 천장 또는 벽에 가까운 정도 등의 환경적 요인이 RSSI 가 들쭉날쭉하게 하여 정확도에 큰 영향을 미치는 것으로 확인되었습니다. 이는 더 오랜 시간 측정을 하거나 혹은 칼만 필터를 제외한 새로운 노이즈 보정 필터가 대두된다면 더 높은 정확도를 얻을 수 있을 것을 기대합니다

개발과정

주간 스프린트 회의



<그림 15> 주간 회의 기록

개발 초기에 개발 환경 구축, 기능 개발, API 연결 등이 병렬적으로 진행되다 보니, 서로의 개발 진행 상황 공유가 원활하지 않은 것을 확인했습니다. 또한, 개발 특성상 개발 중간 결과를 주기적으로 평가하고 그중 가장 적합한 방안을 채택해야 하는 경우가 잦았습니다.

이러한 점을 고려하여 저희는 매주 날짜를 정해 주간 스프린트 회의를 진행하였습니다. 주간 회의에서는 한 주 동안 진행한 내용을 공유하고, 필요한 경우 내부 평가를 진행했습니다. 만약 개발에 문제가 생겨 계획대로 진행되지 않은 파트 또한 공유했습니다. 이후, 다음 주까지 완료해야 하는 마일스톤을 재설정했습니다. 병목 현상으로 인해 기존 계획대로 진행할 수 없는 업무의 경우, 일정을 재조정하여 진행하였습니다.

회의를 진행한 후에는 회의록을 업로드하여 지난 개발 상황을 언제든 확인할 수 있도록 하였습니다.