공학석사 학위논문

단어 임베딩 및 데이터 인코딩을 이용한 다음 시점 사용 모바일 어플리케이션 예측 방법

2022년 1월

서울대학교 대학원

지능정보융합 전공

송태의

단어 임베딩 및 데이터 인코딩을 이용한 다음 시점 사용 모바일 어플리케이션 예측 방법

지도 교수 권가진

이 논문을 공학석사 학위논문으로 제출함
2022년 1월

서울대학교 대학원
지능정보융합 전공
송태의

송태의의 공학석사 학위논문을 인준함
2022년 1월

위 원 장 이원종 (인)

부위원장 권가진 (인)

위 원 이교구 (인)

초 록

다음 시점에 사용 모바일 어플리케이션 예측 문제 Next Application Most Likely Used(이하 NAMLU)는 기계를 이용하여 자동으로 모바일 디바이스에서 사용자가 다음 시점에 사용할 어플리케이션이 무엇인지 예측하는 문제이다. 기존의 문제를 통해 살펴본 문제 해결에 필요한 요소는 세 가지이다. 1) 사용자의 어플리케이션 사용 내역 내의 어플리케이션들 간의 사용된 순서 정보와 단말의 상태 정보를 파악하는 것과 2) 사용자가 어플리케이션을 사용한 시간과 해당 시간에 사용한 어플리케이션 간의 상관 관계를 파악하는 것, 3) 마지막으로 사용자에게 즉각적인 피드백을 줄 수 있도록 빠른 추론 속도를 보장하는 것이다.

본 연구에서는 NAMLU 문제를 해결하고 세 가지 문제를 풀기 위해 TED(Time Series Deep learning) 모델을 제안한다. TED 모델은 단어 임베딩과 시계열 데이터 인코딩 기법을 사용하는 CNN 기반 딥러닝 모델이다. TED의 구성 요소 및 예측 흐름은 다음과 같다. 단어 임베딩을 이용하여 앱 사용 순서 정보와 상호관계를 학습하였다. 이어서 사용자가 어플리케이션을 사용한 시간과 해당 시간에 사용한 어플리케이션 간의 상관 관계를 학습하였다. 이를 위해 임베딩 된 결과를 시계열 데이터 인코딩 알고리즘으로 2차원 메트릭스로 변환한 후 딥러닝 모델에 입력으로 사용하였다. 학습 및 예측을 위한 딥러닝 모델은 사용자에게 즉각적인 피드백을 줄 수 있도록 빠른 추론 속도를 보장하고자 병렬학습이 가능한 CNN 기반 모델을 구현하였다.

본 연구에서는 성능 실험과 사용자 검증 실험을 진행하였다. 성능 실험에서 TED 모델은 Livelab 데이터세트에서 Recall@5 기준 84.5%를 기록하였으며, 이는 동일 데이터세트를 사용한 기존 선행 연구 대비 약 5% 포인트 높은 성능이다. 또한 선행 연구 대비

20% 정도의 파라미터만 사용하였으며, 1초 이상의 빠른 속도를 보일 수 있음도 확인할 수 있었다. 성능 실험에 이어서 NAMLU 문제 해결에 적합한 모델 훈련 방법을 사용자 검증 실험을 통해 확인하였다. 사용자 검증 실험은 학습데이터와 테스트 데이터를 나누는 방식에 따라 크게 4가지 모델 훈련 방법을 나누고 각각을 비교해 보았다. 실험을 통해 확인된 NAMLU 문제에 적합한 모델 훈련 방법은 Transfer learning 방법임을 확인할 수 있었다.

본 연구는 사용자의 어플리케이션 사용 내역과 시간 정보의 중요성을 다시금 확인하였다. 그리고 TED 모델을 사용함으로써 NAMLU 문제에서 Convolution 계열의 모델이 사용 가능하며 모델의 성능과 사이즈 그리고 추론 속도를 고려해 보았을 때 합리적인 선택이 될 수 있음도 확인하였다. 마지막으로 사용자 검증 실험을 통해 NAMLU 문제에 적합한 훈련 방법을 찾은 것에서 연구의 의의를 찾을 수 있었다.

주요어: 다음 시점 사용 어플리케이션 예측(predict next application most likely used), 단어 임베딩(word embedding), 시계열 데이터 인코딩(time series data encoding), 모바일(Mobile), 예측모델(Prediction Model)
학 번: 2020 – 29879

목 차

표 목차

그림 목차

제 1 장 서 론

제 1 절 연구의 배경

스마트폰과 모바일 어플리케이션 사용량은 나날이 증가하고 있으며, 이들은 우리 삶에서 뗄레야 뗄 수 없는 관계가 되었다. 리포트[1]에 따르면 애플 앱 스토어에는 약 220만 개, 구글 플레이 스토어에는 280만개의 다운로드 가능한 어플리케이션들이 등록되어 있다. 또한, 사용자들은 매달 평균 30개의 어플리케이션을 사용하며 60 ~ 90개의 어플리케이션이 사용자의 단말에 설치되어 있다고 한다. 스마트폰에 설치되는 어플리케이션 수가 증가함에 따라 사용자로 하여금 원하는 어플리케이션을 빨리 찾을 수 있도록 도와주며, 이를 빠른 시간 안에 로딩 하는 것은 제조사와 사용자 모두에게 중요한 이슈가 되었으며, 학계에서도 높은 관심을 가지고 다양한 연구가 발표되고 있다.[2]

자동으로 사용자가 다음에 사용할 모바일 어플리케이션(Next App Most Likely Used 이하 NAMLU)을 예측하는 문제는 스마트폰의 대중화와 기계를 이용하여 자동으로 예측하는 방법의 발전에 따라 다양한 연구가 이어졌다. [2, 3] 여기서 NAMLU 문제란 현재까지 사용자가 사용한 앱 리스트를 보고 다음에 사용할 앱이 무엇인지 예측하는 문제다. 예를 들어, 다음과 같이 사용자가 시간 순서에 따라 n−1번째 시점까지 사용한 앱 사용 내역이 있을 때, $\{App_1, App_2, App_3, \dots App_{n-1}\}$ 로 정의된 사용 내역을 보고 n번째 어플리케이션이 무엇인지 예측하는 문제라 할 수 있다.

사용자가 사용할 앱을 미리 예측하는 NAMLU 예측 모델의 구현은 사용자의 단말 사용성 향상 측면과 모바일 운영체제(Operating system)의 시스템 리소스 관리 그리고 배터리 사용량

관리 측면에서도 매우 효과적이다. 먼저, 사용성 측면에서 보았을 때 사용자가 다음 시점에 사용할 확률이 높은 어플리케이션의 예측이 가능하다면 사용 확률을 기준으로 홈스크린 화면을 배치하는 등 모바일 환경에 맞는 새로운 UX/UI를 제안할 수 있다. 또한, 모바일 운영체제의 시스템 리소스 관리 측면에서 볼 때 다음에 사용할 어플리케이션을 예측하여 가장 확률이 높은 어플리케이션을 메모리 상에 올려 둔다면 진입 및 로딩 속도를 향상시킬 수 있으며, 사용할 확률이 가장 낮은 어플리케이션의 리소스를 해지해 둔다면 시스템 리소스를 절약할 수 있다. 마지막으로, 배터리 사용량 관리 측면에도 기여할 수 있다. 사용자가 다음에 사용할 확률이 낮은 어플리케이션을 강제 종료(Force stop) 시킬 경우 백그라운드에서의 예상치 못한 동작을 막을 수 있으며, 사용 확률이 높은 어플리케이션을 확인하여 배터리 사용량을 예측하거나 시스템을 관리하는데 도움을 줄 수 있다.

그림 1. Next App Most Likely Used 문제의 정의

NAMLU 문제를 자동으로 풀기 위해서, 기계는 주어진 어플리케이션 사용 내역에 내포된 어플리케이션들 간의 상호관계와 시간정보를 습득하여 예측을 할 수 있어야한다. 특히, 모바일 사용이 보편화 되면서 연구자들은 자동으로 NAMLU 문제를 해결할 수 있는 모델을 개발하기 위해 노력하였다. Hand-crafted

feature를 이용한 모델[1,2]부터 시작하여 최근에는 자동으로 특성(Feature)을 추출할 수 있게 되었다. 특히, 기계 학습(Machine learning)의 발전을 통해, 일부 연구자들은 뉴럴 네트워크(Neural network)를 사용하여 모델을 만들어 자동으로 특성(Feature)를 추출하고 다음 사용 어플리케이션을 예측하는 문제를 푸는데 기여하였다. [3, 4]

NAMLU 문제를 자동으로 풀기 위해서, 최근의 기계학습 기반의 연구는 다양한 형태의 정보 인코딩 방법을 사용하여 정보를 추출하려는 시도를 하거나 순서가 있는 정보를 학습하기 위해 Recurrent Neural Network 계열의 모델을 사용하는 방법을 시도하였다. 하지만 이러한 방법론들은 다음과 같은 한계점을 지닌다.

먼저, 모바일에 알맞은 정보 인코딩 방법을 제안하지 못하였다. 모바일 사용내역은 단순히 순서의 정보만이 중요한 것이 아니라 순서 정보와 함께 해당 앱 사용 시점에 대한 추가적인 정보가 필요하다. 예를 들어 위치 정보나 시간 정보 혹은 단말의 상태 정보 등이 있다. 이러한 정보는 선행 연구를 통해 모델의 예측 성능을 높일 수 있음이 확인되었다. [1, 2, 5] 특히, 위치 정보와 시간 정보의 특성 중요도(feature importance)가 높음을 확인할 수 있었다.[6] 하지만 위치 정보의 경우 개인 정보 보호에 대한 우려가 있기 때문에 실질적인 사용을 위해서는 어플리케이션 사용 내역과 함께 시간 정보가 함께 반영된 인코딩 방법론이 필요하다. 이때, 어플리케이션 사용 내역 내의 순서 정보와 단말의 상태 정보를 함께 고려하는 방법과 사용한 어플리케이션과 시간 관계를 함께 인코딩 할 수 있어야 한다. 사용자들이 이전에 사용한 어플리케이션이 무엇인지는 다음 어플리케이션을 예측하는데 필수적인 요소이다.

또한, 실질적인 사용을 위해서는 모바일 단말에서 수행되는 모델의 특성을 고려해야 하지만 기존 선행 연구에서는 이러한

부분에 대한 고려가 미약하였다. 특히, 사용 순서에 따라 구성된 어플리케이션 사용 내역을 학습하기 위해 사용되는 선행 연구에서 주로 사용되었던 Recurrent Neural Network는 병렬 학습이 어렵기 때문에 추론 속도가 빠르지 못한 단점을 가지고 있다. 따라서 추론 속도를 향상시킬 수 있는 방법에 대한 고려가 필요하다.

제 2 절 연구의 내용

본 논문에서는 선행 연구에서 한계점으로 확인된 모바일에 적합한 정보 인코딩 방법과 모바일 환경에 적합한 모델구조를 제안하기 위해 시계열 데이터 인코딩 및 단어 임베딩을 이용한 딥러닝 모델, TED(Time series Embedding Deep learning) 모델을 제안한다. 제안된 모델은 선행연구 한계점을 극복하고자 3가지 도전과제를 상정하여 이를 해결함으로써 예측도를 높인 모델이다. 3가지 도전과제는 첫째, 기계는 NAMLU 문제에 적합한 정보 인코딩 방법으로 어플리케이션 사용 내역을 구성하는 어플리케이션들 간의 순서 관계성 및 단말의 상태 정보를 학습하여야 한다. 둘째, NAMLU 문제를 위해 어플리케이션들 간의 상호 관계와 함께 시간 정보도 함께 인코딩 할 수 있어야 한다. 마지막으로, 모바일 단말에서 사용하는 환경을 고려한 빠른 추론 속도를 가질 수 있는 구조를 제안할 수 있어야 한다.

세 가지 도전과제를 다루기 위해서, 우리는 다음 세 가지 기술을 이용하였다. 첫 번째 도전과제를 해결하기 위해 모바일 어플리케이션 사용 내역과 해당 사용 내역과 관련된 단말의 상태 정보를 하나의 문장 형태로 변환하고 이를 단어 임베딩 방법을 사용하여 학습하도록 하였다. 두번째 도전과제를 해결하기 위해 시계열 데이터 인코딩 방법론(time series data encoding

method)[7]를 사용하여 임베딩 된 사용 내역에 시간 정보를 추가할 수 있도록 하였다. 마지막으로 세번째 도전과제를 해결하기 위해 Convolutional Neural Network 기반의 모델을 이용하였다. 이를 통해 병렬 학습이 가능한 구조를 구성하여 Recurrent Neural Network 기반의 모델 대비 빠른 추론 속도를 가질 수 있도록 하였다.

제안된 기술들이 3가지 도전과제를 해결할 수 있는지 학인하기 위해 본 논문에서는 모델의 결과 분석(result analysis)를 진행했다. 실험은 두 가지 실험을 진행하였다. 첫 번째 실험은 NAMLU 문제에 적합한 단어 임베딩 방법과 시계열 데이터의 인코딩 방법을 찾는 실험이다. 문제 해결에 적합한 단어 임베딩 방법과 데이터 인코딩 방법의 조합을 찾고 이들 방법들의 효과를 확인하였다. 두 번째 실험으로는 NAMLU 문제에 적합한 모델 학습 방법을 확인하였다. 사용자 검증 실험을 사용하여 각기 다른 4개의 검증 방법을 사용하였다. 각 실험 방법은 Mixed, User dependent, User independent 그리고 Transfer learning 방법으로 진행하였고, 결과의 분석을 통해 NAMLU 문제에 적합한 모델 학습 방법을 확인하였다. 진행 한 실험의 분석을 통해서 본 연구는 다음과 같은 기여점들을 가진다.

- 첫째, NAMLU 문제에 적합한 데이터 인코딩 방식을 파악하였다. 단어 임베딩 방법과 시계열 데이터의 인코딩 방법을 함께 적용해 어플리케이션 사용 내역을 이루는 어플리케이션들 간의 상호관계 및 시간관계를 동시에 인코딩 하여 NAMLU 문제에 알맞은 데이터 표현(Representation)으로 표상할 수 있었다.

- 둘째, Convolutional Neural Network 기반의 모델 구조와 함께 최적화 기법을 적용함으로써 NAMLU 문제

해결에 적합한 모델 및 학습 구조를 제안하였다.

- 셋째, NAMLU 문제에 적합한 모델 학습 방법을 사용자 검증 실험을 통해서 확인하였다. Mixed, User dependent, User independent 그리고 Transfer learning 방법을 시도해 보았으며 최종적으로 NAMLU 문제에 Transfer learning 방법이 도움이 될 수 있음을 확인하였다.

이어진 논문의 구조는 다음과 같다. 먼저 2절에서 세 가지 도전과제와 관련된 NAMLU 예측 모델 개발을 시도하였던 선행연구들에 대하여 소개할 것이다. 이 후 우리는 3절에서 단어 임베딩과 시계열 데이터의 이미지 인코딩 방법론을 함께 사용한 딥러닝 모델을 제안한다. 본 연구에서 제안하고자 하는 모델의 전체적인 구조와 함께 모델을 이루는 세부 모듈에 대해서 각각 설명하고 해당 모듈의 구현 방안에 대해서도 함께 서술할 것이다. 이어서 4절에서 모델의 성능을 파악하기 위한 실험 세팅을 설명할 것이다. 이때 데이터 세트에 대한 설명과 함께 실험하고자 하는 두 가지 실험에 대한 설명을 서술할 것이다. 이번 연구에서 수행한 두 가지 실험에 대한 결과는 5절에서 논의될 것이다. 각 실험의 결과를 확인하고 결과에 따른 분석을 함께 수행할 예정이다. 마지막으로 6절에서는 실험 결과 및 분석을 바탕으로 한 본 연구의 결론을 이야기할 것이다. 이때, NAMLU 문제를 풀기 위해 설정한 세 가지 연구 목표를 가지고 진행한 실험의 결과에 따른 연구의 의의와 함께 한계점에 대하여 논의할 것이다.

제 2 장 관련 연구

이번 절을 통해 3가지 도전과제와 관련한 선행 연구들에 대해서 논의해보고자 한다. 3가지 도전과제에 대한 선행 연구의 접근법은 다음과 같이 분류해 볼 수 있다. (1) 어플리케이션 사용 정보와 단망상태 정보를 학습하는 방법들, (2) 시간 정보를 학습을 위해 인코딩 하는 방법들, (3) NAMLU 문제를 해결하기 위한 기존의 모델링 방법들, 이상 각각의 방법에 대해서 NAMLU 문제를 다루는 선행 연구 및 NAMLU 문제를 다루지 않더라도 접근 방법과 관련성이 있는 선행 연구를 논해보고자 한다.

제 1 절 어플리케이션 사용 정보와 단말 상태 정보 학습

NAMLU 문제를 풀기 위해 사용되는 어플리케이션 사용 내역의 경우 순서 정보가 존재하기 때문에 선행 연구에서는 이를 학습하고자 하는 시도가 있어왔다. 먼저, hand crafted feature를 이용하여 순서 정보와 이와 관련된 연관 정보를 함께 학습하고자 하는 방법이 있었으며, 임베딩을 이용하여 순서 정보 자체를 학습하고자 하는 경우가 있었다.

먼저, NAMLU 문제를 해결하기 위해 hand crafted feature를 사용한 예측 모델들은 어플리케이션 사용 내역과 해당 시점에서의 단말의 상태 정보를 수동으로 추출하고 학습하였다. 예를 들어 Shin[4]의 연구는 hand crafted feature 들을 카테고리 별로 나누어진 총 37개의 특성(feature)을 이용하여 다양한 기계 학습 모델을 사용해 학습하고 동시에 각 특성의 중요도(feature

importance)를 계산하였다. 이러한 연구를 통해 NAMLU 문제를 다루는 모델의 성능에 영향을 미치는 요소로 위치 정보와 시간 정보가 중요하게 다뤄야 함을 알 수 있었다. 그 외에도 NAMLU 문제를 다룰 때 사용되는 특성으로는 이전에 사용한 어플리케이션들과 단말의 현재 상태 정보가 중요한 특성으로 간주되었다. NAMLU 문제에서 사용된 주요 특성들은 대개 모바일 기기에서 사용되는 센서 정보에 기반한다. 선행 연구 [4, 5, 6]에서 특성 발견(feature discovery) 연구를 통해 확인된 주요 특성은 다음과 같다.

Sensor	**Context**	**Information**
GPS	위도, 경도, GPS 상태 등	위치 정보
Time	요일, 현재 시간 등	시간 정보
Battery	충전기 연결 여부 배터리 잔량 온도 등	배터리 정보
App	사용 앱 내역 앱 상태 (업데이트 여부)	어플리케이션 사용 정보
Network	3G, LTE 여부 Wi-Fi 연결 여부	네트워크 상태 정보
Setting	블루투스 연결 화면 상태 조도 등	단말 설정 정보

표1 NAMLU 문제에서 사용되는 주요 특성들(features)

선행 연구들[2-6]를 통해 확인된 공통적인 주요 특성으로는 시간 정보와 위치 정보 그리고 이전에 사용한 어플리케이션 내역이 있다. 이중에서 위치 정보의 경우 선행 연구에서 주요 인자로써 연구되었다. Dong[7]의 연구의 경우 사용자가 모바일 디바이스를 사용한 장소를 Point-of-interest로 지정하고 주요 위치에 대한 정보를 통신 기지국 정보를 이용해 별도의 데이터 베이스에 관리하고, 각각 장소의 정보와 모바일 디바이스 사용 내역을 연계하여 학습하였다. Huandong[8]은 시공간 정보를 기반으로 어플리케이션 사용 정보를 활용하여 사용자 모델링을 수행하기도 하였다. 그리고 모델링 된 사용자 모델을 이용하여 어플리케이션 사용 예측 및 사용 장소 예측에 활용하였다. T. Do [9]의 연구 역시 사용자가 회사에 있는 경우와 집에 있는 경우 사용되는 어플리케이션이 달라지듯이 위치에 따라 사용되는 어플리케이션이 달라짐에 따라 각 위치에서 사용되는 어플리케이션의 통계를 작성하고 통계에 기반한 모델을 구축하여 장소와 사용 어플리케이션 내역을 함께 예측에 활용 하였다. 이처럼 위치가 미치는 사용 어플리케이션의 영향력은 선행 연구를 볼 때 매우 높음을 알 수 있었으며, 예측 모델에서도 위치 정보는 높은 특성 중요도를 갖는 것을 확인할 수 있었다. 하지만 위치 정보를 주요 인자로 사용하여 예측 모델을 만들 경우 개인 정보 보호에 따른 이슈가 발생할 수 있다. 특히 GDPR(General Data Protection Regulation)[10]과 같은 개인정보 법률에서는 위치 정보를 주요 개인 정보 중 하나로 분류하고 있으며 개인 정보 보호에 대한 중요도가 날로 높아지는 가운데 개인 정보에 민감한 정보를 직접적으로 사용하기에 어려움도 있다.

개인 정보에 민감한 위치 정보가 아닌 중요 특성 중 하나인 시간에 집중한 연구들도 있었다. 사용자가 어플리케이션을 사용한 시간 정보에 집중하여 특정 사용자가 어느 시간에 어떠한 어플리케이션을 사용하는지 확인하는 연구들이다. 일례로 Y.

Tan[11]의 연구는 시간 정보와 함께 네트워크 통신 패킷 정보를 활용하여 사용자의 어플리케이션 사용 내역을 예측하고 학습하였다. 또한 C. Tan[12]의 연구는 직접적으로 시간 대 별 어플리케이션 사용에 대한 통계치를 작성하고 해당 시간에 어떠한 어플리케이션이 사용되었는지 기록하였다. 기록된 데이터를 기반으로 한 통계모델을 바탕으로 예측 알고리즘을 제안하고 이를 어플리케이션 사용 예측 문제에 사용하였다. 이처럼 시간에 집중한 선행 연구들이 있었지만 이들은 대부분 전문가에 의한 수동으로 추출된 특성들을 사용한다는 문제가 있었다. 전문가에 의해 수동으로 추출된 특성을 사용할 경우 데이터의 구조가 변하거나 모델을 업그레이드할 시에 또 다시 전문가의 도움이 필요함에 따라 이에 대한 비용이 발생할 수 있다.

특정 특성에 집중하는 대신에 모바일 단말의 상태정보 전체를 사용하려는 시도들도 있었다. 단말의 전체 상태 정보를 활용하는 것에 집중한 연구로는 Hwang[13]의 연구가 있다. 이 연구는 단말의 상태 정보와 어플리케이션 사용 내역을 모두 하나의 벡터에 표현하고 추천 알고리즘을 사용해서 다음에 사용할 어플리케이션을 예측하고자 하였다. 단말의 상태 정보와 사용 내역을 포함한 벡터는 ATF(App usage Tracking Feature)로 명명하여 위치 정보나 배터리 정보와 같은 명시적 특성들을 함께 학습에 사용해 추천 알고리즘을 구성하였다. 이때, ATF 역시 장소 정보와 함께 사용했을 경우 가장 높은 성능을 보임을 확인할 수 있었다. Y. Xu[14]의 연구에서도 어플리케이션 사용 내역을 App Bag 이라는 동일 공간에 표상하고 이를 단말의 상태 정보와 함께 학습하여 분류기 (Classifier)를 통해 학습하는 모델을 제안하기도 하였다. 이들 선행 연구 역시 수동으로 추출된 특성을 사용한다는 점에서 많은 비용을 발생할 수도 있다는 한계를 지니고 있다.

이러한 한계점을 극복하고자 수동으로 특성을 추출 하는 것이 아닌, 모바일 단말의 상태정보와 어플리케이션 사용 내역을 임베딩

하여 사용 내역 내에 잠재하는(latent) 정보를 학습하고자 하는 시도들도 있었다. 특히 어플리케이션 사용 내역 정보가 자연어와 유사하게 순서가 있는 시퀀스(Sequence) 정보이기 때문에 자연어 처리에서 사용된 임베딩 방법론들이 종종 사용되었다.

최근에 기계 학습 기술의 발전과 함께, 신경망을 이용해서 사람이 레이블을 붙인 별도의 데이터를 활용하지 않고 언어 표상을 학습시키는 비지도 학습(unsupervised learning)을 활용한 방법들이 활발하게 제안되었다[15]. 이러한 연구의 일환으로 언어 정보를 벡터 공간으로 함축적으로 표현하는 분산 언어 표상 연구들이 활발하게 이어져 왔다[16-20]. 일례로 단어 임베딩 기반의 Word2Vec[16]이나 Doc2Vec[17] 과 같은 신경망 기반의 분산 언어 표상 모델들이 임베딩 방법론으로 사용되기도 하였다.

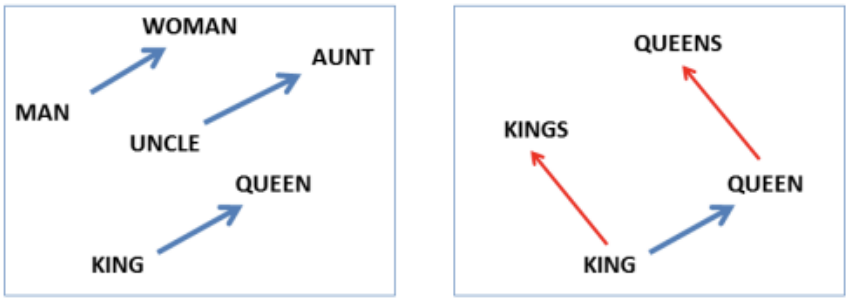

그림 2 단어 임베딩을 사용해 표상한 단어 벡터의 관계 정보

이러한 단어 임베딩 기반의 방법론은 비단 자연어 처리 분야 외에도 추천 시스템과 같이 순서가 있는 정보, 시퀀스 데이터 정보를 표현하는데도 자주 사용된다[21, 22]. 추천 시스템의 아이템을 추천하는 경우 이전에 사용한 아이템의 내역을 확인하고 이를 기반으로 다음 아이템을 추천하는 과정을 기본적으로 거치게 된다[23]. 따라서 단어 임베딩 기반의 방법론을 사용해 이전에 사용된 아이템들의 순서 관계를 벡터 공간에 표상하는 표현 방식을

만들게 된다면 이를 학습에 사용할 수 있게 된다. 추천 시스템 외에도 모바일 데이터와 관련된 연구에서도 벡터 정보를 활용한 연구들이 진행되었다[24]. Feng[25]의 연구는 word2vec[16]을 사용해서 일반적인 어플리케이션 사용 내역을 벡터 공간에 표상하고 일반적이지 않은, 이상적 행동을 보이는 사용 내역을 찾아내어 멀웨어(Malware) 어플리케이션을 찾는 연구를 발표하기도 하였다. 또한, 동일하게 모바일 보안과 관련된 연구 분야로 Le Yu[26]의 연구에서도 단어 임베딩을 사용하여 사용자의 행동을 분석하고 이를 모바일 단말의 보안 정책을 수립하는데 사용한 연구를 선보이기도 했다.

어플리케이션 사용 내역을 직접적으로 다루는 NAMLU 문제와 관련된 선행 연구에서도 단어 벡터를 비롯한 임베딩을 사용한 연구들이 다수 있어 왔다. Zhao[27]의 연구는 Doc2Vec[17]과 어텐션(Attention)을 이용하여 사용 내역을 벡터 공간에 표상하고 이를 어텐션(Attention)이 포함된 심층 신경망을 사용해 학습하도록 하였다. 이 연구에서 제안한 AppUsage2Vec 모델은 어플리케이션 사용 내역을 하나의 문서로 상정하고 문서의 태그(Tag)를 사용자로 두어 Doc2Vec 알고리즘으로 임베딩한 특징을 가진다. 또한, 어텐션(Attention) 방법을 사용해서 이전에 사용한 어플리케이션 중 어떤 어플리케이션이 예측하고자 하는 어플리케이션에 큰 영향을 미쳤는지도 학습할 수 있도록 하였다. 하지만 AppUsage2Vec의 경우 한달이라는 짧은 시간에 대해서만 학습을 진행하였으며, 시간 관계에 대한 임베딩을 가지고 있지 않아 NAMLU 문제에 주요 특성 중 하나인 시간 정보를 모델에 반영하기 어려웠다는 문제를 가지고 있다.

그 외에도 사용자에 따라 어플리케이션 사용 내역을 임베딩하고 이를 학습에 이용하는 다양한 시도의 연구들이 있었다. Chen[28]의 연구와 Zhou[29]의 연구는 장소 단말 상태 정보와 사용 내역을 모두 임베딩 하여 그래프 기반의 모델을 사용하여

학습하는 연구를 소개하기도 하였다. Xiang[30]의 경우 단말의 상태 정보를 비롯한 Context 정보를 벡터화 하여 표상하고 이를 베이지안 모델을 사용하여 학습하기도 하였다.

어플리케이션의 순서 정보와 단말의 상태 정보를 학습하여 NAMLU 문제를 해결하는 다양한 방법의 선행 연구들이 있어왔다. 수동으로 특성들을 찾는 방법에서부터 자연어 처리 방법론을 사용해서 시퀀스 정보 내에 숨겨진 정보를 학습하기 위한 시도들이 있어 왔다. 하지만 두 방법 모두 일부 한계점을 가지고 있다. 먼저, 수동으로 특성을 찾아서 사용하는 방법은 전문가에 의한 추출이 필요하며 이에 따른 확장성의 한계와 비용이 많이 소모된다는 단점이 존재한다. 또한, 자연어 처리 방법론을 사용한 임베딩 방법론의 경우 시간 정보를 정확히 반영하지 못하거나 임베딩 사이즈가 지나치게 커지기에 모델 학습에 필요한 비용과 더불어 임베딩을 구성하기 위한 추가적인 비용이 소모된다는 한계점이 있다. 이에 우리는 어플리케이션 사용 내역과 단말 상태 정보를 함께 학습함과 동시에 임베딩에 소요되는 비용을 최소화하기 위한 데이터 표현 방법을 제안하고자 한다.

제 2 절 사용 어플리케이션과 시간 관계의 학습

시간 정보를 학습하는 인코딩 방법론으로 대표적으로 시계열 데이터의 인코딩 방법론이 있다. 시계열 데이터의 인코딩 방법은 1차원의 시계열 데이터를 2차원의 행렬(matrix)로 변환함으로써 시간 데이터들 간의 상호 관계성과 시간적 관계성을 표현할 수 있는 인코딩 방법론이다. 이러한 시계열 데이터의 인코딩 방법론은 시계열 데이터의 분류(classification) 문제에 많이 응용되고 있다[31-34]. 대표적인 알고리즘으로는 GAF(Gramian Angular Field), MTF(Markov Transition Field) 그리고 RP(Recurrent

Plot)가 있다. 각각 알고리즘은 다음과 같다.

(1) GAF(Gramian Angular Field)는 먼저 식 (1)과 같이 시계열 데이터 X의 원소인 x 들을 0과 1 사이 값으로 정규화 해준다.

$$\tilde{x} = \frac{x - min\,(X)}{(X) - min\,(X)} \qquad (1)$$

그리고 정규화된 x를 극좌표계에 나타낼 수 있도록 식(2)와 (3)을 사용한다. 각각의 식은 극좌표로 표현된 시계열 데이터의 각과 반지름을 나타내며 극좌표계 상에서 시계열 데이터의 변화를 표현해 준다.

$$\emptyset = arccos\,(\tilde{x})\,, -1 \le \tilde{x} \le 1, \tilde{x} \in \tilde{X} \qquad (2)$$

$$\gamma = \frac{t_i}{N}\, t_i \in N \qquad (3)$$

GAF는 식 (2) 와 (3)을 통해 극좌표계에 시계열 데이터를 표현할 경우 직교 좌표계에 표현했을 때와 비교하여 시퀀스 내의 절대적인 시간 관계를 보존할 수 있는 장점을 가질 수 있다[31]. 식 (1) ~ (3)을 통해 극좌표계로 표현된 시계열 데이터는 최종적으로 식 (4) 와 같이 Gramian Matrix 로 표현되어 GAF를 만들어낼 수 있다.

$$G = \begin{bmatrix} cos(\emptyset_1 + \emptyset_1) & \cdots & cos(\emptyset_1 + \emptyset_n) \\ \vdots & \ddots & \vdots \\ cos(\emptyset_n + \emptyset_1) & \cdots & cos(\emptyset_n + \emptyset_n) \end{bmatrix} \qquad (4)$$

GAF 표현된 시계열 데이터는 서로 다른 시계열 데이터 간의 시간적 의존성을 보존할 수 있다는 점에 있어 어플리케이션 사용 내역 내의 어플리케이션 간의 시간 관계를 학습하는데 도움이 될 수 있다.

(2) MTF (Markov Transition Field)는 GAF 와 유사한

인코딩 기법으로, 이산적인 시계열 데이터 간의 전이 확률(transition probability)을 계산한 인코딩 기법이다. MTF는 Markov Transition Matrix를 사용하여 시계열 데이터를 표현하는 방식으로 시간 축을 따라 마르코프 체인(markov chain) 방식을 사용하는 전환 수를 카운팅 하는 기법을 기본 골자로 하여 행렬을 구성한다. 2D 행렬로 변환된 MTF의 각 요소들은 한 시계열 포인트에서 다른 시계열 포인트로의 전환 가능한 우도(likelihood)를 의미하기에 어플리케이션 사용 내역 내의 각 앱들 간의 전환 가능 정도를 학습하는데 도움이 된다 볼 수 있다.

(3) RP(Recurrent Plot)의 주된 아이디어는 시계열 데이터의 궤적이 이전 상태로 돌아가는지를 나타내는 것이다. 시계열 데이터의 궤적이 이전에 방문한 위치로 돌아오는 시간을 측정하는 것으로 시계열 데이터의 공간 궤적을 m차원의 공간 궤적으로 구성한 후 이를 Recurrent Plot 행렬로 표현하게 된다. RP의 경우 이전 위치에 다시 방문하게 되는 경로와 확률 분포를 행렬 내에 포함하기 때문에 어플리케이션 사용 내역 내에서 어플리케이션 사용 패턴이 주기성을 띄게 되는 경우가 많을 때 적합한 인코딩 기법이 될 수 있다.

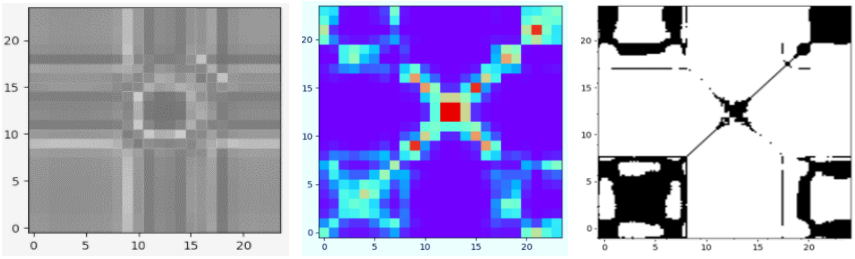

그림 3 시계열 데이터의 이미지 인코딩 예시(GAF, MTF, RP 순)

GAF, MTF 그리고 RP는 시계열 데이터를 다루는 선행 연구 중에서도 시계열 데이터의 분류(Classification) 문제에서 주로 사용되었다. 이때 GAF, MTF 그리고 RP 인코딩 방법만 단독으로

사용되기 보다는 심층 신경망 모델, 그 중에서도 특히 컨볼루션(Convolution) 계열의 심층 신경망 모델과 조합하여 사용되는 경우가 많았다[31-34]. 시계열 데이터의 인코딩 방법을 사용할 경우 1차원의 시계열 데이터가 2차원의 행렬 형태의 데이터로 변환이 됨에 따라 이미지 분류 문제에서 높은 성능을 보였던[35-36] 컨볼루션 계열의 딥러닝 모델과 함께 사용되어 분류 문제의 성능을 높이곤 하였다.

NAMLU 문제의 경우 자연어 처리 문제와 유사하게 어플리케이션의 사용 내역의 순서가 중요하지만 동시에 단순히 순서 외에도 해당 순서에서 사용한 시간 정보 역시 중요 하다는 점이 자연어 처리와 NAMLU 문제와의 다른 점이라 할 수 있다. 하지만 아직까지 시간 정보를 어플리케이션 사용 내역 및 컨텍스트 정보에 중점적으로 인코딩하기 위한 방법은 없었으며, NAMLU 문제를 시계열 데이터로 다루고자 시도한 연구도 현재까지 조사된 바에 의하면 확인할 수 없었다. 따라서 선행 연구에서 확인된 시간 정보의 중요성을 고려해 볼 때 본 절에서 제안하는 시계열 데이터 인코딩 방법과 이전 절에서 제안한 단어 임베딩 방법을 함께 적용할 경우 선행 연구에서 확인하기 어려웠던 시간 관계성을 표현(Represent) 하기 용이할 것으로 예상된다.

제 3 절 NAMLU 예측 모델의 모델링 방법

NAMLU 문제를 해결하기 위해 다양한 모델이 제안되어 왔다. 특히 순서 정보를 가지는 어플리케이션 사용 내역의 특성에 따라 시퀀스 데이터를 분석하고 학습하는데 용이한 것으로 알려진 모델들이 사용되어 왔다. 대표적으로 Recurrent Neural Network 계열이 모델들이 제안되었으며, 마르코프 한 특성을 고려한 모델들과 기계학습 방법을 활용한 모델들도 소개되어 왔다.

먼저, 마르코프 한 특성을 활용한 모델로는 현재의 상태는 바로 직전 상태에 영향을 받는 다는 마르코프 특성(Markov Property)를 응용한 선행연구들이 있다. Huang의 연구[38]의 경우 38 명의 사용자를 대상으로 어플리케이션 간의 전이 확률 행렬을 구축하고, 두 개의 앱 들 간 전이 확률을 계산하는 방식의 마르코프 모델을 사용해 Top-5 어플리케이션 기준 69%의 예측 성능을 기록하였다. 그 외 에도 Y. Xu의 연구[14]와, Parate의 연구[39]도 있었다. 마르코프 특성을 사용한 방법론의 경우 다양한 시도가 있었지만 마르코프 모델의 특성 상 바로 전에 사용한 어플리케이션만 다음에 사용할 애플리케이션에 영향을 준다는 한계점이 있어 직전 시점 이전에 사용되었던 애플리케이션 들과의 관계성을 모델링 하기 어렵다는 문제가 있었다.

또 다른 방식으로는 기계학습 방법을 사용하여 NAMLU 문제를 해결하고자 한 시도들이 있었다. Do의 연구[9]는 다양한 맥락 정보(Context Feature)를 회귀 모델(Regression Model) 및 랜덤 포레스트(Random Forest) 모델을 사용해서 학습하여 NAMLU 문제를 해결하고자 시도되었다. 또한, Xu의 연구[14]는 2단계 분류(Classification) 모델을 제안하였다. 첫 번째 단계에서는 먼저 사용자를 분류하고 이후 두 번째 단계에서 다음에 사용할 어플리케이션을 예측하는 방법을 제안하였다. Xu의 모델은 4600명의 사용자를 대상으로 하였으며 Top-5 어플리케이션 기준 67%의 예측 성능을 보였다. 그 외에도 Xiang의 연구[30]와 같이 베이지안 모델을 사용하여 NAMLU 문제를 풀고자 하는 시도도 있어왔다. 이러한 머신러닝 기반 모델들의 경우 사용자에 의해 추출된 특성(feature)들의 영향을 크게 받는다는 점과 시간 정보와 어플리케이션의 상관 관계를 함께 반영한 데이터 표현 방법을 구현하기 어렵다는 문제가 있었다.

최근에는 NAMLU 문제를 해결하기 위해 다양한 방식으로 딥러닝 모델을 활용하는 방법이 소개되고 있다. 대표적으로

Recurrent Neural Network 기반의 모델을 구현한 모델들을 들 수 있다. Xu[40]의 연구는 Recurrent Neural Network를 직접적으로 사용하여 NAMLU 문제를 해결하려는 시도를 하였다. 해당 연구에서는 어플리케이션 사용 내역을 별도의 인코딩이나 임베딩 없이 LSTM을 사용해서 학습하였으며, 이때 시간 특성과 위치 특성도 함께 활용하여 두 가지 특성이 성능에 미치는 영향을 살펴보기도 하였다.

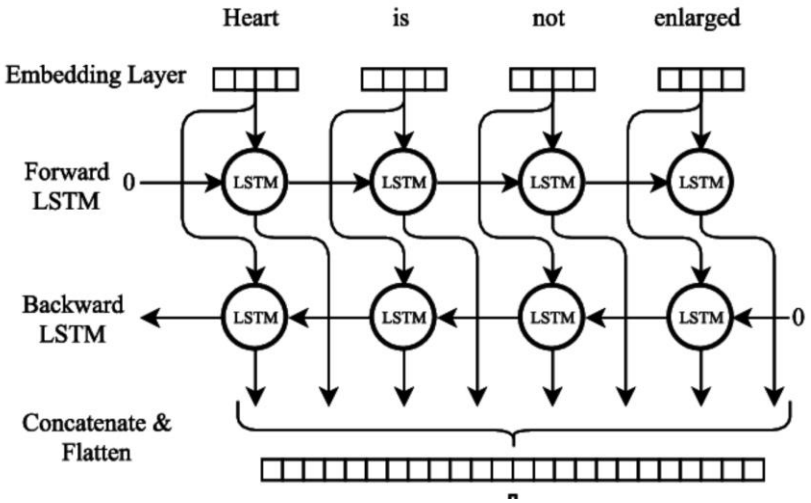

그림 4 Bi-directional LSTM의 구조

Recurrent Neural Network 방법에 기반을 둔 모델 중에서도 Moreira[41]의 연구는 NAMLU 문제에서 가장 좋은 성능을 기록한 모델이다. 해당 연구가 제안한 모델은 NAP(Natural App Processing) 모델로 사용자의 어플리케이션 사용 내역과 단말의 상태 정보를 하나의 문장으로 만들고 Bi-directional LSTM을 사용해 마치 언어 모델(Language Model)을 사용하여 다음 단어를 예측하듯이 문장으로 구성 어플리케이션 사용 내역을 학습하였다. 그리고 문장의 마지막에 나타날 단어를 예측하듯 문장으로 구성된 어플리케이션 사용 내역의 마지막 부분의 어플리케이션을 예측하는 방식으로 NAMLU 문제를 다루었다. 즉, 자연어 처리에서 사용된

기법인 언어모델(Language Model)을 사용해 모델을 구축한 연구이다.

딥러닝 기반 방법론을 사용함에 있어 Recurrent Neural Network에 기반을 둔 방법론만 있는 것은 아니다. 앞서 언급한 Xao[27]의 연구는 Doc2Vec을 이용하여 어플리케이션 사용 내역을 임베딩 하고 Attention이 적용된 Dual Neural Network를 사용하여 학습을 수행하기도 하였다. 또한 강화 학습에 기반한 방법론을 제안한 Xhen[42]의 방법론 또한 있었다. 이렇게 제안된 딥러닝 기반 방법론은 마르코프 특성을 이용한 모델이나 기계학습에 기반한 방법론들에 비해 상대적으로 높은 예측 정확도를 보였으나 그럼에도 불구하고 한계점도 가지고 있다. 지금까지 제안된 딥러닝 모델 기반 연구에서 사용된 데이터 표현방법론으로는 시간 정보와 순서 정보가 가지는 상호 관계성을 동시에 가지기에는 부족하였다. 또한, 실질적인 사용이 고려되어야 할 NAMLU 문제의 특성을 고려하지 않았다. 즉, 빠른 추론 속도나 모바일 단말에서 On-device learning이 수행되어야 하는 특성을 고려하지 않는 모델들이었다. 예를 들어 임베딩 시에 사용되는 임베딩의 차원이 자연어 처리 문제를 풀듯이 100 ~ 200차원 이상을 사용하기 때문에 모바일 단말에 올라가기에는 지나치게 크다는 단점이 있어 왔다. 이와 함께 NAMLU 모델이 주로 Recurrent Neural Network 기반의 모델링을 사용함에 따라 추론 속도에서도 이점을 가지기 어려웠다. Recurrent Neural Network는 구조 상 병렬 처리가 불가능하기 때문에 순차적으로 학습을 수행해야 한다[43]. 이에 Recurrent Neural Network 구조가 아닌 병렬 처리가 가능한 컨볼루션 계열의 모델을 사용하게 될 경우 학습 속도와 함께 추론 속도에서도 이점을 가질 수 있는 장점이 있다. 컨볼루션 계열의 모델의 경우 시퀀스 데이터와 같이 순서 관계가 있는 경우라도 연구 목표나 데이터와 특성에 따라 Recurrent 계열의 모델에 비해서 보다 빠른 속도와 성능의 이점을

가질 수도 있다[44].

이에 본 연구에서는 NAMLU 문제 해결을 위해 다음과 같은 선행 연구의 문제점을 해결하는 것을 목표로 하여 연구 목표를 설정하고자 한다. 어플리케이션의 순서 정보와 컨텍스트 정보를 임베딩하는 방법을 제안하고, 제안된 임베딩 차원을 최소한으로 설정하여 모델의 학습 및 추론 속도 향상에 도움을 주고자 한다. 또한 시간 정보의 중요성을 인식하고 이를 임베딩 차원에 함께 인코딩 할 수 있는 방안을 시계열 데이터의 인코딩 방법을 사용해 마련하고, 나아가 2차원으로 인코딩 된 데이터를 컨볼루션 계열의 모델을 사용해 학습하여 추론 속도를 높이려는 시도를 하고자 한다. 각각에 대한 상세 연구 목표는 다음 장에 서술하였다.

제 3 장 연구 방법

제 1 절 연구의 목표

NAMLU 문제를 해결하기 위해서 심층 신경망 기반의 모델을 제안하고 해당 모델의 성능을 높이기 위해 단어 임베딩 방법과 시계열 데이터의 인코딩 방법의 최적의 조합을 제안한다. 본 연구에서 상정한 연구의 목표는 다음과 같다.

- 연구목표 1) 기계는 NAMLU 문제에 적합한 정보 인코딩 방법을 제안한다. 어플리케이션 사용 내역을 구성하는 어플리케이션들 간의 사용 내역의 순서 관계성 및 단말의 상태 정보를 학습하여야 한다.
- 연구목표 2) NAMLU 문제를 위해 어플리케이션들 간의 상호 관계와 함께 시간 정보도 함께 인코딩 할 수 있도록 하여야 한다.
- 연구목표 3) 모바일 단말에서 사용하는 환경을 고려한 빠른 추론 속도를 가질 수 있는 구조를 제안할 수 있어야 한다.

위와 같이 상정된 연구 목표는 선행 연구의 분석에 따른 결과로 선행 연구에서 확인된 시간이라는 특성의 중요성과 어플리케이션 사용 순서의 중요성을 인식하고, 선행 연구 대비 속도를 향상시킬 수 있는 구조 변경이 가능함을 확인한 결과이다. 본 연구에서는 상정된 세 가지 연구 목표 달성 시에 NAMLU 문제를 해결하고 NAMLU 문제를 다루는 모델의 예측 능력을 향상시킬 수 있으리라 가정하였다.

제 2 절 TED 모델

우리는 본 연구에서 NAMLU 문제의 예측 능력을 높이기 위해 단어 임베딩 방식과 시계열 데이터 인코딩 방법을 순차적으로 적용하고 이를 Convolutional Neural Network 기반의 모델을 사용해 학습하는 TED(Time series Embedding Deep learning) 모델 구조를 다음과 같이 제안하고자 한다.

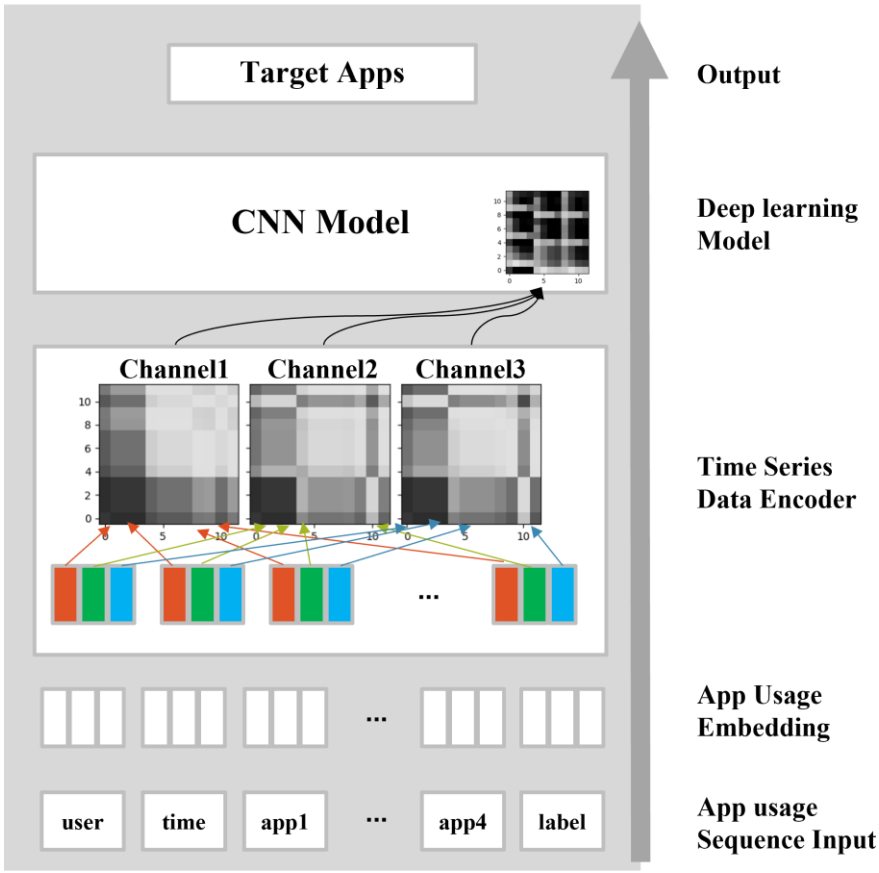

그림 5 TED(Time series Embedding Deep learning) 모델의 구조

TED 모델은 크게 세가지 부분으로 구성되어 있다. 첫 번째 부분은 인풋과 임베딩 부분으로 모델은 인풋으로 어플리케이션

사용 시퀀스(Sequence)를 받아오고 이를 단어 임베딩 알고리즘을 사용하여 벡터로 변환하게 된다. 두 번째 부분은 데이터 인코딩 부분으로 벡터로 변환된 데이터를 2차원의 행렬 형태로 변환해 주는 부분이다. 마지막으로 딥러닝 모델 부분으로 Convolution 계열의 모델을 사용해 다음에 사용자가 사용할 어플리케이션을 학습 및 예측하도록 하였다.

TED 모델을 사용해 NAMLU 문제의 해결 능력을 높이기 위해 풀어야 하는 세 가지 연구 목표에 대한 해결책도 함께 제시한다. 본 연구에서 풀고자 하는 세부 목표 세 가지에 대한 구현 내용은 다음과 같이 세부절에서 설명하려 한다.

먼저 3.1절에서는 데이터 전처리 과정으로 어플리케이션 사용 내역과 해당 시퀀스(Sequence)를 가지는 세션(Session) 에서의 단말의 상태 정보(Context)를 문장화 하고 이를 단어 임베딩 방법을 사용해 임베딩 하는 방식을 설명한다. 또한 데이터 세트를 전처리 과정을 통해 TED 모델 학습에 용이한 형태로 변경하는 방법에 대해서도 서술하였다. 이를 통해 첫번째 도전 과제인 사용자의 앱 사용 순서 정보 내의 앱들 간의 상호관계를 이해할 수 있어야 함을 어떻게 해결하였는지 논한다. 이후 3.2절에서 임베딩 된 데이터를 어떻게 시계열 데이터 인코딩 방식을 적용하였는지 설명한다. 첫 번째 단계에서 임베딩 된 어플리케이션 사용 내역과 단말의 상태 정보는 시계열 데이터의 인코딩 방법론을 적용함으로써 시간 관계성도 함께 인코딩 되는 구조로 구성하였다. 이로써 두 번째 도전 과제인 앱 사용 순서 정보 내의 앱들이 가지는 시간관계성을 이해할 수 있어야 함에 대한 해결 방식을 설명한다. 마지막으로 3.3절에서는 모델 구조 적용 방안에 대하여 설명하고자 한다. 이를 통해 세 번째 도전과제인 실질적인 사용을 위해 모델의 빠른 추론 속도를 가질 수 있는 구조의 구성에 대하여 설명하도록 한다.

3.2.1 사용내역의 문장화와 단어 임베딩

[그림 6]은 전 처리 과정(Preprocessing)을 통해 어떻게 원본 데이터셋을 문장 형태의 데이터로 변환하는 과정을 보여준다. 사용자 별로 시간 순서에 따라 정렬된 데이터를 어플리케이션 사용 세션(Session) 별로 분리하고 각 세션에서 사용자 ID와 시간 정보를 추출한다. 시간 정보로는 사용 요일과 세션 시작 시간, 세션 종료 시간을 사용한다. 이때 세션의 기준은 5개의 어플리케이션 사용 내역을 하나의 세션으로 삼는다. 5라는 값은 다수의 선행 연구[27, 40, 41]에서 실험을 통해 결정된 하이퍼-파라미터(Hyper parameter)다. 실험을 통해 NAMLU 문제에 적합한 사용 내역의 길이를 정한 후에는 전 처리 과정을 수행하여 세션 별로 문장화를 수행하였다.

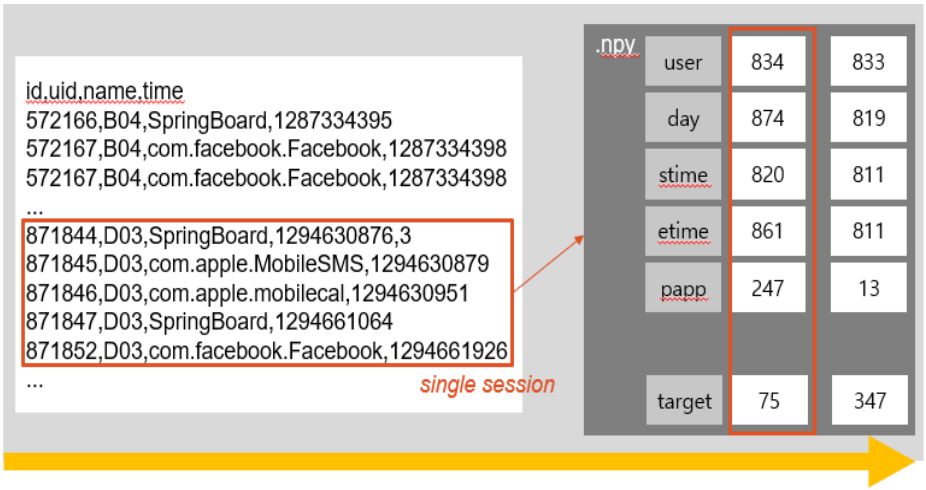

그림 6 전처리 과정을 사용한 사용내역의 문장화

전처리 과정의 세부 내용은 다음과 같은 단계로 수행하였다. 우선 데이터를 사용자 별, 시간 순으로 정렬한 후에 분 단위로 데이터를 병합하였다. 즉, 1분 내에 같은 사용자가 같은

어플리케이션 사용 레코드가 있을 경우 이는 중복된 데이터로 처리하여 삭제하였다. 이후에는 시간 정보에서 사용된 요일 정보와 데이터를 앞서 말한 5개의 어플리케이션 사용 단위로 나누어 5개의 레코드가 하나의 세션(Session)을 구성하 수 있도록 데이터 전처리를 수행하였다. 이때 추가되는 정보로는 시간 정보 외에 각 어플리케이션의 카테고리 정보를 세션에 추가하였다. 사용자 정보와 시간 정보 그리고 어플리케이션의 카테고리 정보와 사용내역이 하나의 세션을 이루도록 구성된다.

세션 별로 데이터를 나누고 나면 각 세션을 하나의 문장으로 간주한다. 이때 세션을 이루는 정보들의 위치 값은 모든 세션에서 동일하게 고정되며 세션의 길이 또한 모든 세션이 동일하다. 즉, 전체 세션의 길이는 동일하며 세션 내의 정보와 어플리케이션 사용 내역 위치도 모두 동일하게 구성하였다. 최종적으로 사용자 id, 요일, 시퀀스 시작 시간, 시퀀스 종료 시간, 이전 어플리케이션 4개의 카테고리, 이전 어플리케이션 4개의 이름 그리고 최종 어플리케이션으로 길이가 13인 세션이 구성되어진다.

{User Id, Day, Start Time, End Time,
0^{th} App's Category, 1^{st} App's Category, …
0^{th} App, 1^{st} App, 2^{nd} App, … N−1th App, Nth App}

이를 통해 마치 "사용자 1은 월요일 13시부터 14시 동안 카카오톡, 페이스북, 크롬, 메시지 앱을 사용하였다" 라는 일정한 템플릿을 가지는 문장을 만들듯이 각 세션을 문장화 한다. 세션의 길이가 13으로 고정되며 이때 마지막은 해당 문장으로 예측하고 하는 타겟 데이터라 할 수 있다. 본 연구에서는 문장 형식으로 구성된 세션 별 정보를 통해 자연어 처리 방법론에서 사용되는 단어 임베딩 방법을 적용해 볼 수 있다고 보았다. 그리고 이를 통해 단말의 상태정보와 어플리케이션 사용 내역을 이루는

어플리케이션들이 이루는 상호 관계성을 학습할 수 있다고 보았다. 또한, 세션을 하나의 문장으로 간주함과 동시에 편의를 위해서 세션의 앞부분, 즉 사용자 정보와 시간 정보가 나타난 부분을 문장에 대한 메타 정보(Meta-information)으로 보기도 하였다. 세션의 앞부분을 메타 정보로 볼 경우에는 이후에 이어지는 카테고리와 어플리케이션 사용 내역은 실제 문장으로 보았다.

단어 임베딩을 적용하기 위해 본 연구에서는 문장화 한 세션을 이루는 각 요소들을 하나의 토큰(Token)으로 간주하고 각각의 토큰을 라벨링 하였다. 이때 라벨링은 메타 정보에 대해서는 임의의 값을 할당하였으며, 어플리케이션 값에 대해서는 데이터 내에서 발생한 빈도 순서에 따라서 값을 할당하였다. 가장 많이 발생한 어플리케이션에 더 적은 값을 할당하고 가장 적게 발생한 어플리케이션에 더 큰 값을 할당하였다. 이때 한가지 고려한 점은 메타 데이터의 라벨 값과 어플리케이션의 라벨 값이 겹치지 않도록 하였다. 데이터 내에서의 발생 빈도에 따라서 값을 할당하였다.

라벨링을 한 후에는 단어 임베딩 방법을 적용하였다. 이 때 본 연구에서는 단어 임베딩 방법론 중 Word2vec 방법[25]과 Doc2vec[26] 방법을 사용하였다. Doc2vec의 경우 두 가지 알고리즘을 가지고 있다. 하나는 PV-DM 알고리즘이며 다른 하나는 PV-DBOW이다. PV-DM 같은 경우 문장 혹은 문서 내의 단어들 간의 순서를 보존하고 임베딩 하는 방법이며, PV-DBOW는 단어 말 뭉치(Bag-of-word) 방식으로 단어들 간의 순서 정보를 고려하지 않은 채 임베딩 하는 방식이다. NAMLU 문제의 경우 어플리케이션들 간의 순서 정보가 중요하기 때문에 PV-DM 방식으로 임베딩을 수행하였으며, 어플리케이션과 카테고리 순서 정보를 각각 문장으로 상정하고, 각 세션의 메타 정보(Meta information)인 사용자 아이디와 시간 정보를 문서의 태그(Tag)로 상정하여 임베딩을 진행하였다. Word2Vec으로 임베딩하는 경우에는 세션 내의 모든 정보를 단어로 상정하여 임베딩을

수행하였다. 이러한 일련의 전처리 과정과 임베딩 과정을 통해 TED 모델의 첫 번째 모듈을 구현하였다. 첫 번째 모듈을 통해 본 연구에서는 NAMLU 과제의 해결을 위한 첫 번째 도전 과제인 앱 사용 순서 정보 내의 앱들 간의 상호관계를 이해할 수 있는 방법을 수행하였다.

3.2.2 임베딩 된 데이터의 시계열 데이터 인코딩

전처리 과정을 거쳐 문장화 된 어플리케이션 사용 내역은 단어 임베딩을 사용하여 각각의 단말 상태 정보와 어플리케이션을 벡터화 되었다. 벡터화 된 각 요소들은 마치 단어가 문장을 구성하듯이 순차적으로 나열되어 있다. 이때, 한가지 고려해야 할 점은 비록 어플리케이션 사용 내역과 시간 정보를 문장화 하였지만 NAMLU 문제를 자연어 처리 방법론만을 사용해 풀 수 없다는 점이다. 선행 연구를 통해 밝혀졌듯이 NAMLU 문제 해결을 위한 주요 인자로는 시간에 대한 속성이 있다. 하지만 단어 임베딩 만으로는 각 어플리케이션 구성 요소가 이루는 시간 관계성을 온전히 학습하기 어려운 문제가 있다. 따라서 시간적 관계성을 데이터 표현 방식에 인코딩 하기 위해 시계열 데이터의 데이터 인코딩 방법을 적용한다.

우선, 임베딩 된 각 벡터들의 하나의 차원이 가지는 값의 변화를 [그림7]에서와 같이 시간 순서에 따라 변화하는 시계열 데이터로 간주한다. 임베딩 된 벡터의 각 차원은 순서대로 값이 변화하는 시계열 데이터로 간주할 경우 각 차원 별로 시계열 데이터의 인코딩 방법을 적용할 수 있다. 데이터의 인코딩은 선행 연구에서 살펴본 MTF, RP, GAF를 사용해서 이루어진다. MTF는 마르코프 성질(Markov Property)에 기반한 값의 변화 확률을 2차원 행렬로 표현하는 방식이며, RP는 시계열 데이터의 변화하는 정도와 다시

값이 돌아올 확률에 대한 행렬 값이다. 마지막으로 GAF는 구현 방식에 따라서 Summation 방법과 Difference 방법 두 가지가 존재하며 각 시간 별 데이터의 상관 관계를 극좌표계를 이용해 표현하고 이를 2차원 행렬 값으로 표현하는 알고리즘이다.

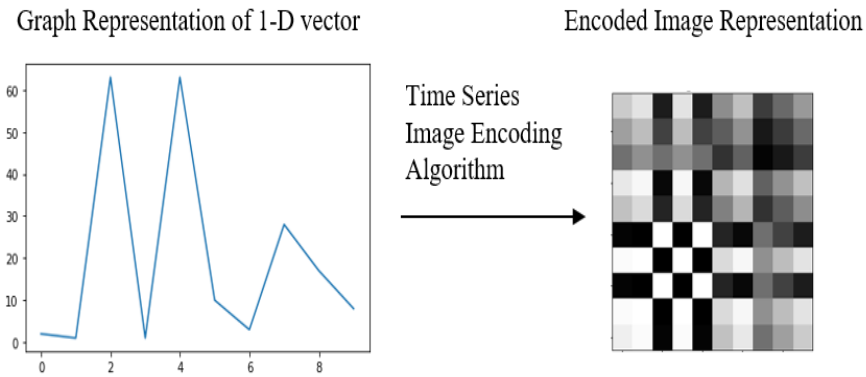

그림 7 순서 및 시간 정보를 가지는 데이터의 시계열 데이터 인코딩

[그림 7]은 시계열 데이터의 데이터 인코딩 방법론을 사용하여 1차원의 데이터를 2차원의 데이터로 변환하는 과정이다. 이때 마치 이미지가 RGB 3개 채널로 이루어져 있듯이 벡터의 각 차원들을 하나의 채널로 간주하고 차원 별로 이미지를 생성하도록 한다. 최종적으로 하나의 문장은 임베딩 차원만큼의 채널을 가지는 하나의 이미지로 변환되도록 구성하였다. 예를 들어 단어 임베딩을 사용해 6차원의 단어 벡터가 만들어졌다면 6개의 채널을 가지는 하나의 이미지 형식의 2차원 데이터가 생성되게 된다. 이를 통해 우리는 NAMLU 문제를 풀기 위한 두번째 도전과제인 앱 사용 순서 정보 내의 앱들이 가지는 시간관계성을 이해할 수 있어야 함을 해결하고자 하였다. 즉, 단어 임베딩이 가지는 어플리케이션들이 이루는 상호 간의 관계성을 임베딩을 통해 먼저 학습하고 이들의 변화하는 정도를 시계열 데이터의 이미지 인코딩 방법론을 적용함으로써 시간에 따른 관계성도 함께 인코딩 될 수 있도록 하였다.

3.2.3 Convolutional Neural Network 모델 구현

마지막 도전 과제인 실질적인 사용을 위해 모델의 빠른 추론 속도를 가질 수 있는 모델을 위해 본 연구에서는 Convolutional Neural Network 기반의 모델을 제안하였다. Convolutional Neural Network의 경우 NAMLU 문제에서 주로 사용되던 Recurrent Neural Network 기반의 방법론과 달리 병렬처리가 가능하기 때문에 보다 빠른 추론 속도를 가질 수 있다.

단어 임베딩과 시계열 데이터의 인코딩 방법까지 거치고 난 후의 데이터는 2차원의 행렬로 벡터의 차원의 수만큼의 채널을 가지는 형식을 띄우게 된다. 이러한 형식의 데이터를 사용함으로써 두 가지 이점을 얻을 수 있도록 하였다.

첫 번째로 이미지 분류에서 높은 성능을 보이는 Convolutional 계열의 모델을 사용할 수 있게 됨으로써 NAMLU 문제를 이미지 분류 문제로 치환하여 각각의 인코딩 된 데이터를 기반으로 다음에 사용할 어플리케이션을 예측할 수 있도록 모델을 구현하였다. 이미지 분류 문제로 치환하기 위해서 시계열 데이터의 인코딩 방법을 사용해 변환된 어플리케이션 사용 내역은 각각 마지막에 사용된 어플리케이션의 아이디로 라벨링 되게 된다. 따라서 NAMLU 문제를 분류(Classification) 문제의 일환으로 풀이할 수 있다. 두 번째로 Convolutional 계열의 모델을 사용함으로써 추론 속도의 향상을 가져올 수 있다. RNN 계열의 모델들에 비해 Convolutional 계열의 모델이 가질 수 있는 장점 중 하나인 병렬처리의 가능성을 높일 수 있다. 또한, 고정되어 있으며 비교적 짧은(12 step) 시퀀스 길이를 가질 경우 한 번에 데이터를 살필 수 있는 Convolutional 계열의 모델이 성능상 이점을 가질 수도 있게 된다.

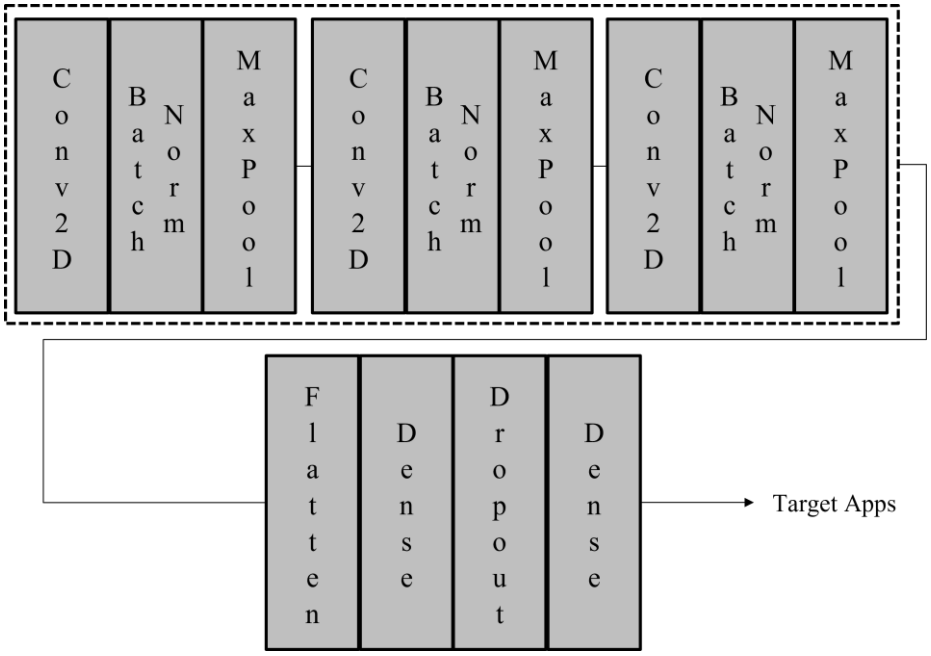

그림 8 TED CNN 모듈 구조

TED에서 사용하는 Convolution 모듈은 Convolution 2D 레이어와 Batch Normalization, MaxPooling2D으로 구성된 Convolution 모듈을 3층으로 쌓고 이어서 Flatten 레이어와 선형 레이어를 두어 2차원의 행렬로 변환된 어플리케이션 사용 내역을 Convolution 모듈이 학습할 수 있도록 구성하였다.

Convolution 모듈의 중간에 Batch Normalization을 추가함으로써 정규화를 통해 학습 데이터에 과도하게 학습되는 현상을 막고 모델의 일반화 성능을 높게 유지[46]하고자 하였다. 그리고 Convolution2D 레이어를 사용함으로써 행렬로 변환된 데이터의 가로 세로 정보 및 채널 정보까지 함께 학습하여 의미 있는 특성을 뽑아내고 이를 최종적으로 학습 및 예측할 수 있는 구조를 구성하였다.

제 4 장 실험계획

이번 섹션에서, TED 모델의 성능을 측정하기 위한 실험 계획과 NAMLU 문제에 적합한 학습 방법을 논하기 위한 실험 계획 그리고 데이터 세트에 대해서 소개를 한다. 먼저, 다음 사용 모바일 어플리케이션 예측 모델인 TED의 성능을 평가하기 위해 사용된 데이터 세트는 공개된 벤치마크 데이터 세트인 Livelab[45]을 사용한다. 그리고 첫 번째 실험은 TED 모델과 다른 선행연구 State-of-the-art 모델을 비교하기 위한 실험을 진행하고자 하며, 실험을 통해 TED 모델에 가장 적합한 임베딩 방법과 인코딩 방법을 확인하려 한다. 두 번째 실험으로는 TED 모델과 NAMLU 문제에 적합한 모델 학습 방법을 사용자 검증 실험을 통해 확인해 보고자 한다. 데이터 세트와 각 실험 방법에 대한 세부 내용 및 계획은 다음과 같다.

제 1 절 데이터 세트

실험에서 사용한 데이터 세트인 Livelab 데이터 세트는 스마트폰 사용자 34명의 약 6개월 간의 어플리케이션 사용 기록을 로깅한 데이터 셋이다. 사용자들은 데이터 로깅 기간 동안 평균적으로 약 140개의 어플리케이션을 사용했으며 최소 40개에서 최대 535개의 어플리케이션을 사용하였다. 총 레코드 수는 572,165 건이다. 성능측정 실험에서는 전체 사용자에 대해서 학습을 진행하였으며 시간 순으로 정렬된 데이터를 7:3의 비율로 학습 세트와 테스트 세트 비율로 나누어 실험을 진행하였다. Livelab 데이터 세트는 딥러닝 모델의 학습에 충분한 양이며 해당

데이터 세트를 이용함에 있어 모델의 성능을 평가하기에 적합한 데이터셋이기에 본 연구의 실험 데이터로 선정하였다.

Livelab 데이터의 Raw 데이터는 어플리케이션 사용 내역 외에도 배터리 사용 정보 및 어플리케이션 별 카테고리 정보 그리고 단말 상태 정보 등이 다수의 테이블로 구성되어 있다. 이 중 어플리케이션 사용 내역은 다음과 같이 구성되어 있다.

id, uid, name, time
572166, B04, SpringBoard, 1287334395
572167, B04, com.facebook.Facebook, 1287334398
572168, B04, com.facebook.Facebook, 1287334601
...
871844, D03, SpringBoard, 1294630876,3
871845, D03, com.apple.MobileSMS, 1294630879
871846, D03, com.apple.mobilecal, 1294630951
871847, D03, SpringBoard, 1294661064
871852, D03, com.facebook.Facebook, 1294661926
...

표2 Livelab의 원본 데이터

각 레코드는 모두 4개의 컬럼으로 구성되어 있다. 먼저 각 레코드의 아이디가 구성되어 있으며, 사용자 별 아이디가 다음 컬럼으로 정의되어 있다. 이어서 사용자가 사용한 어플리케이션의 이름 혹은 패키지 명이 정의되어 있다. 마지막으로 사용자가 어플리케이션을 사용한 시점에 대한 시간 정보가 작성되어 있다. Livelab Raw 데이터는 어플리케이션의 카테고리와 시간 정보에서 추출한 요일 정보 및 시(hour) 정보를 이용하여 하나의 세션 별 상태 정보 및 순서 정보를 문장화 형식으로 전 처리해 학습에 사용하였다. Livelab 데이터 세트는 각 테이블 별로 NAMLU 문제에 적합한 정보가 구성되어 있기에 연구를 위해 선정할 만한

데이터 세트라 할 수 있다.

Livelab 데이터 세트는 정보 공개 측면에 있어서도 연구를 위해 선정될 만한 데이터 세트이다. NAMLU 문제를 풀기 위해 사용된 데이터 세트의 대부분은 공개되어 있지 않는 데이터 세트인 경우가 많다. 이는 모바일 단말의 사용 내역과 단말의 상태 정보 등 개인 정보의 많은 부분을 공유해야 하는 특성이 있기 때문이다. 최근의 연구에서는 대개 통신사와 연구자가 협업하여 통신사로부터 비공개 데이터를 제공받아 연구를 진행하는 경우가 많았다. 이에 반해 Livelab 데이터 세트의 경우 개인 정보를 제외한 NAMLU 문제를 풀기에 적합한 수준의 정보가 공개된 데이터라 할 수 있다.

Livelab 데이터는 원본 데이터의 경우 약 57만 건의 데이터 레코드를 가지고 있으며 총 34명의 사용자가 총 743개의 어플리케이션 조합을 약 6개월 동안 사용한 데이터 세트이다. 실험에 앞서 세션 별로 전처리가 완료된 데이터의 개수는 약 20만 건으로 전 처리가 완료된 데이터를 이용하여 TED 모델을 학습하도록 시험을 구성하였다.

Category	Value
User	34
Original Records	572,165
Preprocessed Records	199,191
Total # of Apps	743
Recorded duration	6 months
Collected By	Rice University

표3 Livelab의 데이터 정보

제 2 절 모델 성능 측정 실험 계획

TED 모델의 성능 측정에 앞서 TED 모델의 구성 요소인 임베딩 모듈과 인코딩 모듈의 효과를 측정하기 위한 Ablation Study를 진행한다. Ablation Study를 통해 각 모듈의 효과를 확인 및 분석한 후에 베이스라인 모델들과 성능 비교를 위한 지표를 측정하였다.

비교하기 위한 베이스라인 모델로는 MRU(Most Recently Used), MFU(Most Frequently Used)와 기존의 선행 연구에서 사용되었던 Recurrent Neural Network 기반 모델인 NAP 모델[41]을 베이스라인으로 삼았다. 특히, NAP는 본 연구에서 사용한 Livelab 데이터를 이용해서 연구를 진행하였으며, 전처리 과정에서 어플리케이션 사용 내역을 문장화 하였다는 점에서 공통점을 가지고 있어 주요 비교 모델로 선정하였다. 베이스라인으로 삼은 MRU와 MFU[47] 알고리즘은 모바일 단말에서 메모리 관리를 위한 LRU Caching[48]과 더불어 메모리 관리를 위해 자주 사용되는 대표적인 알고리즘 중 하나이다. 각각의 이름에서 직관적으로 알 수 있듯이 MRU는 가장 최근에 사용된 어플리케이션을 반환하는 알고리즘이며, MFU는 최근에 사용한 어플리케이션 중 가장 자주 사용한 어플리케이션을 반환하는 알고리즘이다.

실험에서 모델들을 비교하기 위해 사용된 지표로는 Recall@K를 사용하였다. Recall@K를 사용하여 베이스 라인 모델과 본 연구에서 제안한 모델의 학습 및 추론을 통해 예측한 어플리케이션들의 상위 K에 정답이 속해 있는지를 판단하여 모델의 성능을 측정하도록 하였다. Recall@K의 경우 비교 모델로 사용한 NAP에서도 동일하게 사용되고 있어 직접적인 비교가 가능하였다. Recall@K의 경우 선행 연구[27, 41]에서 사용되던 지표로 지표의

정의는 다음과 같다.

$$Recall@K = \frac{\sum_{i}^{|D^{Test}|} 1(P_{aj} \geq P_{[k]})}{|D^{Test}|} \quad (5)$$

위 식은 다음과 같이 정의된다. 우선 분모의 경우 전체 테스트 데이터의 개수를 의미한다. 그리고 분자의 $1(*)$ 은 지시함수(Indicator function) 이며, 지시함수 내의 조건 중 전자는 정답 어플리케이션을 의미하고, 후자의 경우 K 번째 높은 확률을 가지는 어플리케이션의 확률을 의미한다. 따라서 지표를 기준으로 다음과 같이 식을 계산할 수 있게 된다.

예를 들어 테스트 데이터의 개수가 10개라 가정하고 각각의 테스트 데이터의 정답, 즉, 시퀀스에서 최종적으로 사용자가 사용할 어플리케이션, NAMLU 어플리케이션이 하기와 같이 정의되어 있다고 한다. 이때 각 숫자는 각 어플리케이션을 의미한다고 가정한다.

[13, 86, 13, 330, 75, 13, 13, 90, 357, 75]

그리고 모델을 통해서 구해진 전체 어플리케이션의 확률 값, 즉 NAMLU 일 확률을 내림 차순으로 정렬하여 가장 높은 5개(K=5)를 구한 값이 하기와 같다고 가정한다.

테스트 데이터1의 예측 값: [77 83 86 **13** 75]
테스트 데이터2의 예측 값: [87 247 75 **86** 13]
테스트 데이터3의 예측 값: [87 86 77 **13** 75]
테스트 데이터4의 예측 값: [88 247 75 86 13]
테스트 데이터5의 예측 값: [84 87 247 **75** 13]

테스트 데이터6의 예측 값: [83 87 86 75 13]
테스트 데이터7의 예측 값: [83 90 247 87 13]
테스트 데이터8의 예측 값: [86 84 87 75 13]
테스트 데이터9의 예측 값: [247 84 87 75 86]
테스트 데이터10의 예측 값: [247 83 86 75 13]

위의 예제 상황을 볼 때 테스트 데이터1, 2, 3, 5, 6, 7 그리고 10번의 경우 모델이 예측한 가장 확률이 높은 5개(K=5) 어플리케이션 리스트 내에 정답 어플리케이션을 포함하고 있다. 그러나 테스트 데이터 4, 8 그리고 9번의 경우 모델이 예측한 가장 확률이 높은 5개(K=5) 어플리케이션 리스트 내에 정답을 포함하지 않고 있어 식(5)와 같이 정의된 Recall@K는 최종적으로 70%를 결과로 가지게 된다. 본 연구에서 사용된 모델의 성능 지표는 선행 연구에서 사용된 Recall@K 즉, 식(5)를 사용하여 측정되었다.

추가적으로 예측성능 뿐만 아니라 추론 속도에 대한 고려도 하여야 하기 때문에 모델들 간의 추론 속도 역시 비교하였으며, 적합한 Convolution Neural Network 구조를 찾기 위해 선행 연구에서 사용되었던 구조를 확인하고 각 모델의 파라미터 수와 속도를 종합적으로 고려하였다. 이와 함께 최적의 조합을 찾기 위한 실험도 진행되었다. 단어 임베딩 방법으로는 Word2Vec과 Doc2Vec 방법이 실험에 사용되었다. 그리고 시계열 데이터의 데이터 인코딩 방법은 MTF, RP, GAF 방법이 사용되었다. 이들의 조합을 확인하고 하이퍼-파라미터 튜닝을 통해 TED 모델에 적합한 구조도 확인하였다.

제 3 절 사용자 검증 실험 계획

성능 측정 실험을 통해 TED 모델에 가장 적합 한 구조와

모듈의 구성 요소를 확인한 후에 사용자 검증 실험을 진행한다. 사용자 검증 실험은 학습 데이터와 테스트 데이터를 사용자에 따라 분리하는 방법으로 나뉘며 4가지 방식으로 구성되어 있다. 사용자 검증 실험의 방식은 사용자 행동 분석(User behavior analysis) 연구에서 주로 사용되는 방식[49, 50]을 차용하였다.

첫 번째 방식은 Mixed 방식이다. Mixed 방식은 사용자의 구분 없이 학습 데이터와 테스트 데이터를 구성하는 방식으로 실험 1을 Mixed 방식으로 진행한다. 두 번째 방식은 User dependent 방식이다. 한 명의 사용자로만 학습 데이터와 테스트 데이터를 구성하는 방식으로 전체 사용자 34명에 대해서 각각 실험을 진행하며, 전체 사용자에 대한 평균을 먼저 구하고 가장 성능이 좋았던 5명과 가장 성능이 좋지 못했던 5명의 평균을 각각 구한다. 세 번째 방식은 User independent 방식이다. 타겟 사용자를 지정한 후에 타겟 사용자를 제외하고 학습 데이터를 구성한 후에 타겟 사용자 만으로 테스트 데이터를 구성하는 실험이다. 마지막은 Transfer-learning 방식이다. 마지막 방식은 타겟 사용자를 지정함에 있어 User independent 방식과 동일하나, 타겟 사용자의 데이터를 한 번 더 재-학습을 위한 데이터와 테스트 데이터로 나눈다는 점에서 차이가 있다. 먼저, 학습 데이터로 사전 학습(Pre-training) 한 후에 사전 학습으로 완성된 모델을 재-학습 데이터로 모델의 가중치(weight) 값을 미세 조정(fine-tuning) 하여 학습하고 테스트 데이터로 모델의 성능을 측정하고자 한다. 각 실험들에 대한 결과와 분석은 다음 장에서 논의한다.

제 5 장 실험결과 및 분석

제 1 절 모델 성능 측정 실험

성능 측정 실험은 Ablation study를 통해 TED 모델의 각 모듈의 효과를 먼저 확인한다. 이어서 하이퍼 파라미터 튜닝(Hyper parameter tuning) 및 임베딩과 인코딩 방법에 따른 조합을 확인하고, 베이스 라인 모델과 성능 및 속도 측정 비교 실험을 진행한다. 각각의 실험 결과는 다음과 같다.

5.1.1 Ablation Study

Models	**Recall@5**
Vanilla	75.1
TED_embedding only	81.4
TED_data-encoding only (GAF)	81.7
TED	**84.5**

표4 Ablation study 결과

Livelab 데이터의 전처리를 완료하고 난 이후 데이터는 각 세션별로 단말의 상태 정보와 어플리케이션 정보가 숫자로 라벨 인코딩된 벡터의 형태를 지닌다. 따라서, Vanilla 모델, 즉 임베딩과 데이터 인코딩이 모두 적용되지 않은 모델은 2D Convolution 이 아닌 1D Convolution을 사용해서 학습해야 하는 형태를 지니게 된다. 따라서, Vanilla 모델의 경우 1D Convolution 모델[50]을 사용해서 학습을 진행했으며 Recall@5 지표 기준 75.1%의 성능을 보임을 알 수 있었다.

이어서 첫 번째 연구 목표인 어플리케이션 사용 내역을 구성하는

어플리케이션들 간의 순서 관계와 단말의 상태 정보를 학습하기 위해 사용된 단어 임베딩 기법의 효과를 확인하였다. 이를 위해 TED 모델에서 임베딩만 적용한 모델의 성능을 확인하였다. 임베딩이 적용될 경우 임베딩의 차원의 수와 세션의 길이만큼의 2차원으로 구성된 행렬이 생성이 되기에 TED 모델 본연의 2D Convolution 모델을 사용할 수 있었다. 임베딩 만 적용한 모델일 지라도 Vanilla 모델 대비 약 6% 포인트 성능이 향상됨을 확인할 수 있었다. 이는 연구 목표로 설정된 어플리케이션 간의 순서 관계와 단말의 상태 정보 학습이 효과적임을 확인한 결과라 볼 수 있다. 단말의 상태 정보와 어플리케이션의 순서 내역의 임베딩 결과에 대해서는 5.1.2절에서 보다 자세히 논하도록 한다.

이어서 TED를 구성하는 두번째 모듈인 시계열 데이터 인코딩 모듈의 성능을 시험하고자 하였다. 임베딩이 적용되지 않는 모델의 경우 어플리케이션 사용 내역을 문장화 한 후에 문장 내 토큰이 라벨링 된 1차원의 데이터를 시계열 데이터의 이미지 인코딩 방법을 이용해 2차원의 1개의 채널을 가지는 이미지로 인코딩하는 방법을 적용하였다. 라벨링 된 토큰을 그대로 시계열 데이터화 하는 것은 자칫 어플리케이션들이 대소 관계를 가지게 된다고 볼 수도 있으나, 이를 GAF나 MTF 혹은 RP와 같은 시계열 데이터 인코딩 방법을 사용하여 인코딩 함으로써 대소 관계를 학습하는 것이 아닌 시간 순서에 따른 토큰들의 변화량을 학습할 수 있도록 유도하였다.

이때, 이미지 인코딩 알고리즘으로는 GAF를 사용하였다. 이는 GAF가 다른 시계열 데이터 인코딩 알고리즘 대비 높은 성능을 보임에 따라 결정되었다. GAF는 두 가지 구현 방식으로 나뉘며 Summation 방법과 Difference 방법이 있으나 두 구현 방식에 따른 큰 차이는 확인할 수 없었다. 각 데이터 인코딩 방법이 TED 모델에 미치는 영향에 대해서는 단어 임베딩 방법과 데이터 인코딩 방법의 조합 결과를 논하는 5.1.3절에서 서술할 예정이다. 결론적으로 데이터 인코딩 알고리즘 만으로도 Vanilla 모델 대비 6% 포인트

성능이 높아진 것을 확인할 수 있었다.

최종적으로 임베딩 모듈과 데이터 인코딩 모듈을 모두 포함한 TED 모델을 사용하여 학습한 경우에는 Vanilla 모델과 비교해 보았을 때 9% 포인트 이상의 성능 향상을 보였으며, 임베딩 모듈이나 시계열 데이터 인코딩 모듈만 적용한 각각의 경우와 비교했을 때도 약 3% 포인트 성능이 높아짐을 알 수 있었다. 이를 바탕으로 단어 임베딩 모듈과 시계열 데이터 인코딩 모듈을 모두 포함할 경우 모델의 성능 향상을 도모할 수 있음을 확인할 수 있었다.

5.1.2 임베딩 시각화

임베딩의 결과를 확인하기 위해 시각화도 진행해 보았다. 시각화는 Tensorflow Embedding Projector[51] 서비스를 이용하였으며 이때 사용된 알고리즘은 UMAP[52]이다. 시각화의 결과는 다음과 같다.

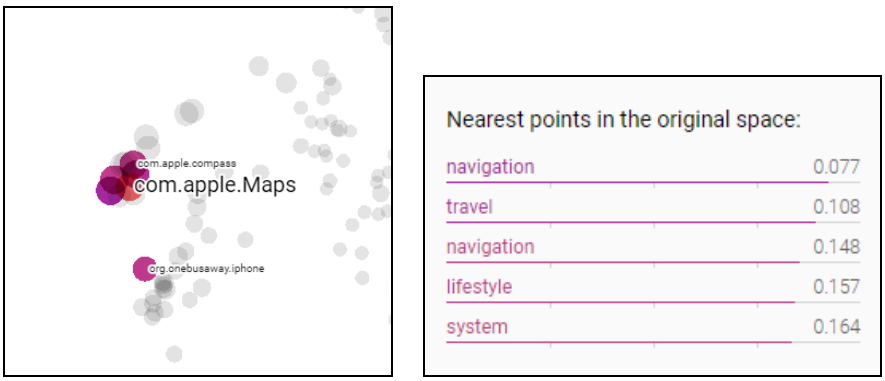

그림 9 지도 어플리케이션의 임베딩 시각화 및 유사 어플리케이션의 카테고리 리스트

상기의 경우 지도 어플리케이션의 임베딩 결과로써, 지도 어플리케이션과 가장 가까운 거리, 즉 가장 유사한 벡터를 가지는 어플리케이션의 카테고리는 네비게이션, 여행, 라이프-스타일 그리고 시스템 어플리케이션 순으로 확인되었다. 지도 어플리케이션의 특성을

생각해 볼 때 여행과 네비게이션 어플리케이션이 높은 유사도를 가짐은 임베딩의 결과가 유의미하게 이뤄졌음을 의미한다고 볼 수 있다. 이외에도 전화기 어플리케이션이나 사진 편집 어플리케이션을 살펴볼 때 유사한 어플리케이션 혹은 관련도가 높은 어플리케이션이 의미공간 상 가까운 위치에 표상됨을 확인할 수 있었다.

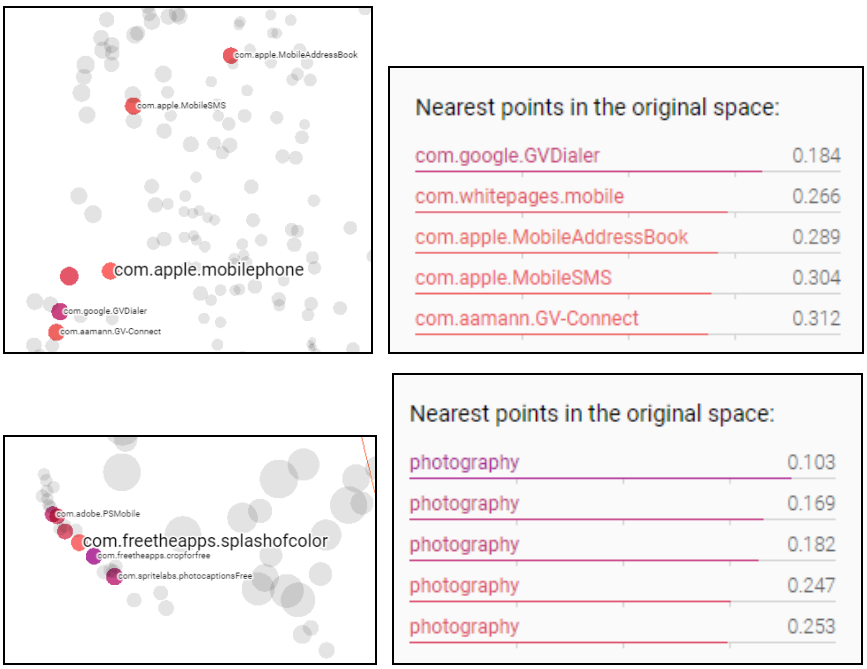

그림 10 전화기 어플리케이션과 사진 편집 어플리케이션의 임베딩 시각화 및 유사 어플리케이션 및 카테고리 리스트

전화기 어플리케이션과 가까운 거리의 어플리케이션으로는 다이얼러 어플리케이션, 전화번호부 문자 어플리케이션 등이 확인되었으며, 사진 편집 어플리케이션과 같이 전문성을 띄며 사용 영역이 특화된 어플리케이션의 경우 모두 같은 유사도를 가지는 카테고리로 묶이는 것을 확인할 수 있었다. 시각화를 통해 어플리케이션 임베딩이 유의미 하게 이뤄졌음을 확인할 수 있었고, 실제 모델 학습에 활용될 경우 학습에 도움이 됨도 알 수 있었다.

5.1.3 단어 임베딩과 데이터 인코딩 조합 확인

Models		Recall@5
Word2Vec		
TED	+ RP	79.1
TED	+ MTF	80.9
TED	+ GAF(DIFF)	83.7
TED	**+ GAF(SUM)**	**83.9**
Doc2Vec		
TED	+ RP	78.2
TED	+ MTF	81.6
TED	+ GAF(DIFF)	84.1
TED	**+ GAF(SUM)**	**84.5**

표5 단어 임베딩 및 데이터 인코딩 조합 별 Recall@5 값 결과

Ablation study 실험에 이어서 NAMLU 문제와 TED 모델에 적합한 단어 임베딩 방법과 시계열 데이터 인코딩 방법의 조합을 확인해 보았다. 단어 임베딩 방식은 Word2Vec 과 Doc2Vec을 각각 사용해 보았다. 그리고 시계열 데이터의 인코딩 방식으로 RP, MTF 그리고 GAF(DIFF), GAF(SUM) 방식을 비교해 보았다. 이를 통해 살펴본 결과는 다음과 같다. 우선 단어 임베딩 방식에 따른 성능 상의 큰 변화는 없었다. 단어 임베딩 방식의 경우 Word2Vec과 Doc2Vec 이 순서 정보를 학습하는 방식이 유사하기 때문에 임베딩이 달라지는 것에 의한 변화는 미비하였다. 이미 고정된 길이의 고정된 자리 값을 가지는 시퀀스이기 때문에 유사한 두 가지 임베딩 방법으로는 성능 상의 큰 변화를 가져오기 어려웠다. 이에 반해 시계열 데이터의 인코딩 방법에 따라서는 성능 상의 차이가 두드러짐을 확인할 수 있었다. 이는 각 시계열 데이터 인코딩 알고리즘의 구현 방식이 NAMLU 문제와 어떻게 상응하는지에 따라 달라진 것으로 보인다.

시계열 데이터 인코딩 방법으로 MTF, RP 그리고 GAF를 비교하였다. GAF의 경우 극좌표계에서 반지름과 각도를 이용해 시간 관계를 표현할 때 합(Summation)을 이용하는 방법과 차(Difference)를 이용하는 방법으로 나뉘어지기에 각각을 나누어 비교하였다. 비교한 알고리즘 중에서도 GAF의 성능이 다른 두 가지 알고리즘에 비해 약 3 ~ 6% 포인트 높은 성능을 보임을 알 수 있었다. 이때 GAF 방식이 합인 경우와 차인 경우 모두 다른 이미지 인코딩 방법론에 비해 높은 성능을 보임을 알 수 있었다. 이는 GAF의 경우 시계열 데이터를 이루는 시점들 간의 시간 관계성을 표현하며, 시간 종속성에 대한 정보가 남아 있도록 인코딩 되는 방법론이기 때문에 어플리케이션 사용내역을 다루는 NAMLU 문제에 있어서 적합한 방식이라 볼 수 있기 때문으로 여겨진다.

이에 반해 RP의 경우 Ablation Study의 Vanilla 모델의 성능과 유사하게 나타나 오히려 임베딩과 함께 사용될 경우 성능 향상이 아닌 하락에 영향을 끼치는 것으로 보인다. 이는 RP의 경우 위상공간 궤도를 탐색하며 어떤 궤적이 이전 상태로 돌아가는지를 들어내는 알고리즘으로 임베딩으로 벡터화 된 값의 궤적이 이전 궤적으로 돌아가는 정도를 파악하기 어렵기 때문으로 여겨진다.

마지막으로 MTF의 경우 인코딩에 따른 효과가 RP에 비해서는 크지만 Ablation study에서 임베딩만 적용한 경우와 비교해 보았을 때 큰 효과를 거둔다고 보기 어려워 보인다. 이는 특정 값의 변화하는 확률을 마르코프 특성(Markov Property)로 표현하는 알고리즘의 특성 상 벡터의 차원 전이 확률을 구하는 것이 NAMLU 문제와 맞지 않기에 발생한 현상이라 여겨진다.

결론적으로 TED 모델을 구성하는 알고리즘으로 단어 임베딩 방법은 Doc2Vec을 사용하였다. 그리고 시계열 데이터의 인코딩 방법으로는 GAF를 사용하였으며, Summation 방법과 Difference 방법 중에서도 Summation 방법을 사용하도록 모델을 구성하여다.

5.1.4 베이스라인 모델과의 비교

앞선 5.1.1절과 5.1.2절을 통해 TED 모델 내의 각 모듈의 효과를 확인하고 최적의 조합을 찾을 수 있었다. 이어지는 실험으로는 베이스 라인 알고리즘인 MFU와 MRU 그리고 동일한 데이터 세트에서 State-of-the-Art의 성능을 보인 NAP 모델과 비교해서 상대적으로 높은 성능을 보일 수 있는지 알아보며, TED 모델에 적합한 Convolution 구조를 확인하기 위해 선행 연구와 비교도 함께 진행하였다. Convolution 구조와의 비교를 위해 사용된 선행 연구는 VGG16[53] 과 MobilenetV2[54]를 사용하였다. VGG16의 경우 파라미터의 수 즉, 모델 사이즈가 LSTM 계열의 State-of-the-Art 모델인 NAP와 비교해 유사하기 때문에 선정하였으며, MobilenetV2의 경우 경량화 된 모델로 모바일 환경에서 높은 성능을 보이고 있기에 선정하였다. 실험의 결과는 아래와 같다.

Algorithm		**Recall@K**
MFU	Most Frequently Used	28.6 (K=1)
MRU	Most Recently Used	31.3 (K=1)

Architecture	**Model**	**Recall@K**
LSTM base	NAP	78.9 (K=5)
CNN base	VGG16	79.2 (K=5)
	MobileNetV2	82.7 (K=5)
	TED	**84.5 (K=5)**

표6 베이스라인 모델과의 비교 결과

실험에서 확인한 결과 TED 모델은 MFU, Most Frequently Used와 MRU, Most Recently Used와 같은 기본 알고리즘 대비 높은 성능을 보임을 알 수 있었다. MFU와 MRU는 가장 자주 사용

한 한 앱, 가장 최근에 사용한 앱 하나씩 만을 선정하는 알고리즘 이기에 명확한 비교를 위해 Recall@1으로 TED 모델도 측정하였으며, TED 모델의 Recall@1 값은 44.7로 약 16% 포인트 높은 것을 확인할 수 있었다.

또한, LSTM이라는 RNN 계열의 모델을 가지고 State-of-the-art를 달성한 NAP 모델과 비교해 보았을 때도 약 6% 포인트 높은 성능을 거두었음을 알 수 있었다. 이는 베이스라인 모델들과 비교해 보았을 때 TED가 어플리케이션 사용 순서 및 단말의 상태 정보를 시간 관계 정보와 함께 인코딩 시킴으로써 보다 효율적으로 모델을 활용하고 있다고 볼 수 있을 것이다.

이어서 같은 Convolution 계열의 모델과의 비교를 위해 VGG16과 MobilenetV2를 사용해 NAMLU 문제를 풀어보고자 하였다. 실험의 결과를 살펴보았을 때 VGG16의 경우 NAMLU 문제를 다루기에는 모델의 사이즈가 상대적으로 크거나 모델의 깊이가 깊어 효율적인 학습이 어려운 것으로 분석되었다. 또한 MobileNetV2의 경우 Depth-wise convolution 레이어를 사용함에 따라 공간적 관계만 학습하게 되고, 채널에 대한 학습은 이와 분리하여 별도로 학습되기 때문에 학습에 적합하지 않았다. 이에 반해 TED의 Convolution 모델 모듈의 구조는 3-depth의 CNN 레이어와 Batch-norm을 가지며, 2차원 행렬의 공간적 정보와 채널 정보를 모두 동일한 커널에 함께 학습할 수 있기에 임베딩의 차원 수만큼 채널로 변환하여 시간 정보를 학습하는 TED 구조에 적합하며, 모델 사이즈 역시 NAMLU 문제를 풀기에 적합한 구조로 확인되었다.

5.1.5 모델 사이즈 및 추론 속도 비교

첫 번째 실험의 마지막 단계로 선행 연구들과 TED 모델의 추론 속도 및 모델 사이즈를 비교해 보았다. 모델 사이즈는 파라미터 수

를 기준으로 측정하였으며 추론 속도는 동일한 테스트 세트를 사용하여 모델이 예측하는 시간을 배치(Batch) 단위로 측정 시 밀리초 단위로 측정하고, 전체 테스트 데이터를 대상으로는 초 단위로 측정하였다. 모델 사이즈 측정 및 속도 측정 실험에 대한 결과는 다음과 같다. [표 7]의 결과에서 볼 수 있듯이 TED 모델의 파라미터 수가 가장 적으며 추론 속도 역시 가장 빠른 것을 확인할 수 있었다. 이를 통해 TED 모델을 사용하여 빠른 추론 속도를 가지게 하는 연구 목표를 확인할 수 있었다.

Model	**Parameters**	**Batch base Inference Time**	**Total Inference Time**
NAP	10,609,841	62.6ms	4.5s
VGG16	15,087,828	102.2ms	7.3s
MobilenetV2	3,538,984	83.2ms	6.4s
TED	**1,962,716**	**48.8ms**	**3.5s**

표7 TED 모델 파라미터 수와 속도 비교 결과

5.1 절의 실험을 통해 본 연구에서는 세 가지 연구 목표 세가지를 달성하였음을 확인하였다.

- 1) 어플리케이션 사용 내역을 구성하는 어플리케이션들 간의 사용 내역의 순서 관계성 및 단말의 상태 정보를 학습하는 것.

- 2) 시간 정보도 연구 목표 1)의 정보에 함께 인코딩 할 수 있도록 하는 것.

- 3) 모바일 단말에서 사용하는 환경을 고려한 빠른 추론 속도를 가질 수 있는 구조를 제안할 것.

이들 연구 목표를 달성함으로써 베이스 라인 모델에 비해 적은 파라미터 수로 높은 성능을 가지는 모델을 구축하 수 있음을 실험을 통해 확인할 수 있었다. 이어지는 실험에서는 첫 번째 실험에서 구축된 모델을 기반으로 사용자 검증 실험을 진행하여 NAMLU 모델에 적합한 모델 학습 방법을 탐구해 보고 결과에 대해 분석해 보고자 한다.

제 2 절 사용자 검증 실험

	Train Data	**Test Data**	**Performance (Recall@5)**	
Mixed	All	All	**84.5**	
User-dependent	Target User	Target User	All	82.4
			top-5	91.3
			bottom-5	75.2
User-independent	All except Target User	Target User	All	81.5
			top-5	89.2
			bottom-5	74.9
Transfer-learning	All except Target User	Target User	**All**	**85.6**
			top-5	**93**
			bottom-5	**78.9**

표8 사용자 검증 실험 결과

실험에서 사용된 Livelab 데이터에는 총 34명의 사용자가 있으며, 34명의 사용자 전체에 대해서 사용자 검증 실험을 수행하였다. 사용자 검증 실험에 사용된 방법은 세 가지이다. 앞서 언급한 사용자 검증 실험 방법으로 Mixed, User dependent, User independent 그리고 Transfer learning 방법이 있으며 이중 Mixed 방법의 경우

사용자 구분 없이 학습 데이터와 테스트 데이터를 구분하는 방식으로 1 절에서 수행한 결과이다. 이에 따라 Mixed 외의 세가지 방법에 대한 실험을 진행하였으며, 이에 대한 결과는 다음과 같다.

User dependent 방법을 사용해 실험을 진행할 경우 개별 사용자의 데이터만을 가지고 학습 데이터와 테스트 데이터를 구성하게 된다. TED 모델을 사용해서 User dependent 방법으로 학습을 할 경우 전체 34명의 사용자의 평균 모델 성능은 Recall@5 기준으로 82% 정도의 성능을 보이며 가장 높은 성능을 보인 5명의 평균만 취했을 경우에는 91% 그리고 가장 낮은 성능을 보인 사용자 5명의 평균을 취했을 경우에는 75%의 성능을 보임을 알 수 있었다. 개별 사용자를 대상으로 모델 학습을 진행할 경우에는 데이터 세트가 충분하거나 데이터에 노이즈가 적을 경우 학습이 용이하지만 반대의 경우에는 매우 낮은 성능을 보일 수 있음을 알 수 있었다.

이어서 진행된 User independent 방법의 경우 타겟 사용자를 테스트 데이터로 구성하고 그 외의 전체 사용자를 학습 데이터로 두어 학습하는 방식이다. User independent 방법을 사용했을 때 전체 사용자의 평균 81.5%로 User independent 방법에 비해 소폭 감소하였으며, 최상위 성능 5명과 최하위 성능 5명의 지표 역시 하락함을 알 수 있었다. 다만, 전체적인 하락의 정도가 크지 않음에서 볼 때 다른 사용자의 학습 데이터가 한 사용자 데이터 학습에 도움이 될 수도 있음을 알 수 있었다.

마지막으로 수행한 사용자 검증 실험은 Transfer learning 방법이다. NAMLU 문제에 적합한 문제를 찾고자 User-dependent 방법과 User-independent 방법을 각각 실험해 보았을 때 다른 사용자의 데이터를 효율적으로 이용할 수 있을 때 모델 학습 성능을 올릴 수 있음을 예상할 수 있었다. 이에 User independent 와 동일하게 학습 데이터와 테스트 데이터를 구분하되 테스트 데이터에 사용하는 타겟 사용자의 데이터를 재학습을 위한 데이터와 테스트를 위한 데이터로 한 번 더 구분하여 다른 사용자로 학습된 모델을 타겟

데이터의 일부로 재 학습 후 평가하였다. 이에 따른 결과로 전체 사용자에 대한 평균을 비롯해 모든 지표가 3%~4% 포인트 상승한 것을 확인할 수 있었으며 Mixed 방법에 비해서도 지표가 상승한 것을 확인할 수 있었다. 이는 전이 학습(Transfer learning)[55]의 효과로 다른 사용자의 데이터를 통해 선 학습한 후에 타겟 사용자의 데이터를 이미 학습된 가중치(weight) 위에 한 번 더 학습함으로 NAMLU 모델에 맞는 학습 방법을 찾을 수 있었다.

제 6 장 결론 및 한계점

본 연구는 NAMLU 문제를 해결하기 위한 3가지 도전과제를 해결하기 위해 TED(Time Series Embedding Deep learning) 모델을 제안하고 실험을 통해 모델의 성능을 확인할 수 있었다. 임베딩과 시계열 데이터의 데이터 인코딩 방식을 순차적으로 적용하여 모델이 어플리케이션 사용 내역을 이루는 어플리케이션들 간의 상호 관계성과 시간적관계성을 모두 학습할 수 있도록 하였다. 단어 임베딩 방법을 사용하여 단말의 상태 정보와 사용자가 사용한 어플리케이션 사용 내역 정보를 임베딩 하여 학습하도록 하였다. 그리고 시계열 데이터의 데이터 인코딩 방식을 사용하여 어플리케이션 사용 기록과 시간과의 관계를 인코딩하여 학습에 활용하였다.

각각의 효과는 Ablation Study를 통해 확인하였으며, 단어 임베딩과 시계열 데이터의 데이터 인코딩이 모두 없는 모델에 비해서 단어 임베딩과 시계열 데이터의 데이터 인코딩이 한 가지라도 적용된 경우의 성능이 각각 약 6% 포인트 높았으며, 두 가지 경우가 모두 적용된 경우에는 9% 포인트에 가까운 성능 향상을 보임을 알 수 있었다. 추가적으로 빠른 추론 속도를 가질 수 있도록 CNN 계열의 모델을 제안하였다. 일련의 방법을 통해 구현된 모델은 Recall@5 지표를 기준으로 하여 84.5%의 성능을 기록하였으며 이는 state-of-the-art 모델인 NAP 모델에 대비해서 약 5% 포인트의 성능 향상을 보인 기록이었다.

모델의 성능에 대한 실험에 이어서 NAMLU 문제에 적합한 학습 방법에 대한 실험 또한 진행되었다. 사용자 검증 실험을 통하여 Mixed-model, User-dependent model, User-independent model 그리고 Transfer learning model을 실험하였다. 결과적으로 Transfer learning model 방식을 사용한 학습 방법이 다른 방식에 의한 모델 성능을 최대 5% 포인트 이상 상회하였다.

제 1 절 연구의 의의

본 연구에서는 연구 목표로 설정된 세 가지 목표인 1) 어플리케이션 사용 내역을 구성하는 어플리케이션들 간의 사용 내역의 순서 관계성 및 단말의 상태 정보를 학습하는 것. 2) 시간 정보도 연구 목표 1)의 정보에 함께 인코딩 할 수 있도록 하는 것. 3) 모바일 단말에서 사용하는 환경을 고려한 빠른 추론 속도를 가질 수 있는 구조를 제안할 것을 달성할 경우 모델의 성능을 향상시킬 수 있을 것이라 예상하였다. 이러한 예상은 실험 1. 을 통해 확인되었으며 이에 따른 연구의 의의는 다음과 같다.

첫 째로 선행 연구에서 기 확인되었던 어플리케이션 사용 내역과 단말의 상태 정보 그리고 시간 관계의 중요성을 다시 한번 확인할 수 있었다. 특히 이들을 개별적으로 모델에 사용하는 것이 아닌 통합적으로 정보를 인코딩 할 수 있을 경우 모델의 성능이 개별 사용 대비 향상됨을 확인할 수 있었다. 이에 따라 중요도가 높은 개별 특성을 보다 효율적으로 인코딩 할 수 있는 방안이 지속 연구될 경우 NAMLU 문제를 해결하는 모델의 예측력은 보다 상승할 수 있을 것으로 보인다.

두 번째로 Convolutional 계열의 모델이 NAMLU에서 활용될 수 있음을 확인하였다. 모델의 성능 향상과 더불어 추론 속도 향상을 위해 기존 선행 연구에서 주로 사용되던 RNN 계열의 모델이 아닌 Convolutional 계열의 모델을 사용하였고, 결과적으로 RNN 계열의 모델 대비 높은 성능과 빠른 추론 속도 그리고 적은 모델 사이즈를 얻을 수 있었다. NAMLU 문제의 경우 대개 살펴볼 시퀀스의 길이가 자연어 처리나 오디오 신호 처리에 비하여 짧으며 고정적인 길이를 가지고 있을 수 있다. 따라서 이러한 경우에는 선행 연구의 실험에서 보이듯 Convolutional 계열의 모델이 대안이 될 수도 있다.

세 번째로 NAMLU 문제에 적합한 학습 방법을 확인할 수 있었다. 실험 2)를 통해 사용자 검증 실험을 수행하였고 이에 대한 결과로 NAMLU 모델에는 Transfer learning 방식이 학습이 도움이 될 수 있음을 확인하였다. Livelab 데이터는 사용자의 수가 실제 환경에 비해 매우 적어 실제 모바일 사용 환경을 반영하기 어렵다는 단점이 있지만 적은 사용자의 데이터를 수집하였기 때문에 사용자 검증 실험에 용이하다는 장점이 있다.

사용자 검증 실험의 결과 Transfer learning 방법이 다른 방법인 Mixed, User dependent 그리고 User independent 방법에 비해 높은 성능을 보일 수 있었던 것은 학습에 사용된 데이터 세트 내의 사용자의 앱 사용 확률 분포 혹은 데이터의 확률 모델이 유사하기 때문에 가능 한 것으로 보인다. 이는 User independent 방법과 User dependent 방법의 실험 결과가 크게 차이 나지 않는 것에서 예측할 수 있다. 이는 다른 사용자의 어플리케이션 사용 내역이 특정 사용자의 NAMLU 예측에 도움이 될 수 있음을 의미한다 볼 수 있다. 따라서, 보다 규모 있는 데이터를 통해서 User independent 방법과 User dependent 방법의 차이가 크게 나는 경우에 Transfer learning 방법의 NAMLU 에서의 효과를 논의할 필요성은 있다. 하지만 그럼에도 불구하고, Transfer learning 방법이 NAMLU 문제를 다룸에 있어서 도움이 될 수도 있음을 확인하였고, 오히려 전체 사용자들을 대상으로 학습하는 것에 비해 효율적일 수도 있음을 확인했다는 점에서 연구의 의의가 있다 할 수 있다.

제 2 절 연구의 한계점

본 연구에서 제안한 모델이 선행 연구 대비 높은 성능을 기록하긴 했지만 여전히 추가적인 실험과 한계점도 존재하고 있다. 먼저, 본 실험에서 사용한 임베딩 기법의 한계이다. Word2Vec 방법이나

Doc2Vec 방법과 같은 단어 임베딩 방법은 자연어 처리 문제를 해결할 때 동음이의어를 처리할 수 없다는 문제가 있다. 자연어 처리와 달리 하나의 어플리케이션이 다수의 어플리케이션 사용 내역에 등장할 수 있는 NAMLU 문제의 특성상 동음이의어를 처리할 수 있는 인코딩 방법론이 필요하다 할 수 있다. 예를 들어 자연어 처리 방법론에서 사용되었던 GloVe[56]와 같은 방법론이나 대규모 언어 모델을 사용한 방법론[57, 58] 등이 예시가 될 수 있다.

또한, Cold start 문제에서도 자유롭지 못한 단점이 있다. 앱의 설치와 제거가 종종 발생하는 모바일 단말 사용 특성 상 정적인 임베딩 방법으로는 새로이 추가된 어플리케이션을 학습하기 어렵다는 단점이 있다. 이로 인하여 학습 및 추론에 있어서도 on-line learning의 어려움이 있을 수 있다.

마지막으로 사용자 검증 실험의 결과에서 논의하였듯 데이터 세트 내의 사용자의 수가 적어 대규모 사용자를 대상으로 한 모델의 성능 및 학습 방법에 대한 확인이 필요하다. 데이터 세트 내의 사용자들 간의 데이터 분포가 유사한 상황이기에 다양한 종류의 데이터 분포를 가지는 대규모 데이터를 통해 동일한 성능과 학습 방법이 가능함을 확인해야 하는 것이 추후 연구에서 필요하다고 볼 수 있다. 혹은 사용자의 국적이나 연령대가 비슷할 경우 Livelab 데이터와 유사하게 비슷한 사용자 별로 Transfer learning 방법 기반의 학습을 진행할 수 있을 것이다. 따라서, 유사한 사용자들을 함께 묶을 수 있는 사용자 모델링 방법과 TED로 증명한 모델 및 학습 방법이 함께 이뤄질 경우 NAMLU 모델의 성능을 보다 올릴 수 있을 것이다.

참고 문헌

1. Mobile App Download Statistics & Usage Statistics (2021) Available online: https://buildfire.com/app-statistics/ (accessed on 12/23/'21)

2. Cao, Hong, and Miao Lin. "Mining smartphone data for app usage prediction and recommendations: A survey." Pervasive and Mobile Computing 37 (2017): 1-22.

3. Zhao, Sha, et al. "User profiling from their use of smartphone applications: A survey." Pervasive and Mobile Computing 59 (2019): 101052.

4. Shin, Choonsung, Jin-Hyuk Hong, and Anind K. Dey. "Understanding and prediction of mobile application usage for smart phones." Proceedings of the 2012 ACM Conference on Ubiquitous Computing.

5. Baeza-Yates, Ricardo, et al. "Predicting the next app that you are going to use." Proceedings of the eighth ACM international conference on web search and data mining. 2015.

6. Liao, Zhung-Xun, et al. "On the feature discovery for app usage prediction in smartphones." 2013 IEEE 13th International Conference on Data Mining. IEEE, 2013.

7. Yu, Donghan, et al. "Smartphone app usage prediction using points of interest." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 1.4 (2018): 1-21.

8. Wang, Huandong, et al. "Modeling spatio-temporal app usage for a large user population." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 3.1 (2019): 1-23

9. T. M. T. Do and D. Gatica-Perez, “Where and what: Using smartphones to predict next locations and applications in daily life,” Pervasive and Mobile Computing, vol. 12, pp. 79–91, 2014.

10. General Data Protection Regulation (GDPR) – Official Legal Text Available online: https://gdpr.eu/ (accessed on 12/23/’21)

11. Tan, Yaowen, et al. "Predicting app usage based on link prediction in user-app bipartite network." International Conference on Smart Computing and Communication. Springer, Cham, 2017.

12. Tan, Chang, et al. "Prediction for mobile application usage patterns." Nokia MDC workshop. Vol. 12. 2012.

13. Hwang, Yongkeun, Donghyeon Lee, and Kyomin Jung. "Mobile app recommendation with sequential app usage behavior tracking." Journal of Internet Technology 20.3 (2019): 827-837.

14. Y. Xu, M. Lin, H. Lu, G. Cardone, N. Lane, Z. Chen, A. Campbell, and T. Choudhury, “Preference, context and communities: a multi-faceted approach to predicting smartphone app usage patterns,” in ISWC 2013. ACM, 2013, pp. 69–76.

15. Bengio, Yoshua, et al. "A neural probabilistic language model." Journal of machine learning research 3.Feb (2003): 1137-1155

16. Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

17. Le, Quoc, and Tomas Mikolov. "Distributed representations of sentences and documents." International conference on machine learning. PMLR, 2014.

18. Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In EMNLP, 1532–1543.

19. Collobert, R.; and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In ICML. 160–167.

20. Mnih, A.; and Hinton, G. 2009. A scalable hierarchical distributed language model. Advances in NIPS, 1081–1088

21. Ozsoy, Makbule Gulcin. "From word embeddings to item recommendation." arXiv preprint arXiv:1601.01356 (2016).

22. Zheng, Lei, Vahid Noroozi, and Philip S. Yu. "Joint deep modeling of users and items using reviews for recommendation." Proceedings of the tenth ACM international conference on web search and data mining. 2017.

23. Das, Debashis, Laxman Sahoo, and Sujoy Datta. "A survey on recommendation system." International Journal of Computer Applications 160.7 (2017).

24. Sen, Sevil, and Burcu Can. "Android Security using NLP Techniques: A Review." arXiv preprint arXiv:2107.03072 (2021).

25. Y. Feng, L. Chen, A. Zheng, C. Gao, and Z. Zheng. Ac-net: Assessing the consistency of description and permission in android apps. IEEE Access, 7:57829–57842, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2912210.

26. Le Yu, Xiapu Luo, Chenxiong Qian, Shuai Wang, and Hareton KN Leung. Enhancing the description-to-behavior fidelity in android apps with privacy policy. IEEE Transactions on Software Engineering, 44(9):834–854, 2017.

27. Zhao, Sha, et al. "AppUsage2Vec: Modeling smartphone app usage for prediction." 2019 IEEE 35th International Conference on Data Engineering (ICDE). IEEE, 2019.

28. Chen, Xinlei, et al. "CAP: Context-aware app usage prediction with heterogeneous graph embedding." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 3.1 (2019): 1-25.

29. Zhou, Yifei, Shaoyong Li, and Yaping Liu. "Graph-based Method for App Usage Prediction with Attributed Heterogeneous Network Embedding." Future Internet 12.3 (2020): 58.

30. C. Xiang, D. Liu, S. Li, X. Zhu, Y. Li, J. Ren, and L. Liang, “Hinextapp: A context-aware and adaptive framework for app prediction in mobile systems,” in 2017 IEEE Trustcom/BigDataSE/ICESS, pp. 776–783.

31. Wang, Zhiguang, and Tim Oates. "Encoding time series as images for visual inspection and classification using tiled convolutional

neural networks." Workshops at the twenty-ninth AAAI conference on artificial intelligence. Vol. 1. 2015.

32. Qin, Zhen, et al. "Imaging and fusing time series for wearable sensor-based human activity recognition." Information Fusion 53 (2020): 80-87.Conference Location:El Paso, Texas USA

33. Barra, Silvio, et al. "Deep learning and time series-to-image encoding for financial forecasting." IEEE/CAA Journal of Automatica Sinica 7.3 (2020): 683-692.

34. Jastrzebska, Agnieszka. "Lagged encoding for image-based time series classification using convolutional neural networks." Statistical Analysis and Data Mining: The ASA Data Science Journal 13.3 (2020): 245-260.

35. K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 2016, pp. 770–778.

36. Albawi, Saad, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network." 2017 International Conference on Engineering and Technology (ICET). Ieee, 2017.

37. Li, Yandong, Z. B. Hao, and Hang Lei. "Survey of convolutional neural network." Journal of Computer Applications 36.9 (2016): 2508-2515.

38. Huang, Ke, et al. "Predicting mobile application usage using contextual information." proceedings of the 2012 ACM conference on ubiquitous computing. 2012.

39. A. Parate, M. B¨ohmer, D. Chu, D. Ganesan, and B. M. Marlin, “Practical prediction and prefetch for faster access to applications on mobile phones,” in UbiComp 2013. ACM, 2013, pp. 275–284.

40. Xu, Shijian, et al. "Predicting smartphone app usage with recurrent neural networks." International Conference on Wireless Algorithms, Systems, and Applications. Springer, Cham, 2018.

41. Moreira, Gabriel S., Heeseung Jo, and Jinkyu Jeong. "NAP: Natural App Processing for Predictive User Contexts in Mobile Smartphones." Applied Sciences 10.19 (2020): 6657.Conference Name:ACM Woodstock conference

42. Shen, Zhihao, et al. "Deepapp: a deep reinforcement learning framework for mobile application usage prediction." Proceedings of the 17th Conference on Embedded Networked Sensor Systems. 2019.

43. Sak, Hasim, Andrew W. Senior, and Françoise Beaufays. "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." (2014).

44. Bai, Shaojie, J. Zico Kolter, and Vladlen Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling." arXiv preprint arXiv:1803.01271 (2018).

45. Shepard, Clayton, et al. "LiveLab: measuring wireless networks and smartphone users in the field." ACM SIGMETRICS Performance Evaluation Review 38.3 (2011): 15-20.

46. Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." International conference on machine learning. PMLR, 2015.

47. LruCache. Available online: https://developer.android.com/reference/android/util/LruCache (accessed on 12/23/'21)

48. So, Kimming, and Rudolph N. Rechtschaffen. "Cache operations by MRU change." IEEE Transactions on Computers 37.6 (1988): 700-709.

49. Martín, Alejandro G., et al. "A survey for user behavior analysis based on machine learning techniques: current models and applications." Applied Intelligence (2021): 1-27.

50. Yoon Kim. 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

51. Smilkov, Daniel, et al. "Embedding projector: Interactive visualization and interpretation of embeddings." arXiv preprint arXiv:1611.05469 (2016).

52. McInnes, Leland, John Healy, and James Melville. "Umap: Uniform manifold approximation and projection for dimension reduction." arXiv preprint arXiv:1802.03426 (2018).

53. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

54. Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

55. Torrey, Lisa, and Jude Shavlik. "Transfer learning." Handbook of research on machine learning applications and trends: algorithms, methods, and techniques. IGI global, 2010. 242-264.

56. Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.

57. Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

58. Radford, Alec, et al. "Improving language understanding by generative pre-training." (2018).

Abstract

Predicting Next Application Most Likely Used using Word Embedding and Time-Series Data Encoding

Taieui Song

Department of Transdisciplinary Studies

The Graduate School

Seoul National University

Predicting Next Application Most Likely Used (NAMLU) is a task that automatically predicts a mobile application that most likely is used next using a machine. There are three factors needed to solve the task examined through previous studies. 1) Identifying usage order information and device state information between applications in the user's application usage history. 2) Identifying the correlation between the time the user used the application and the application used. 3) Ensuring fast inference speed to provide immediate feedback to the user.

In this paper, a Time Series Deep Learning (TED) model is proposed to solve the NAMLU problem and satisfy those three factors. The TED model is a CNN-based deep learning model that uses word embedding and time-series data encoding techniques. The components and prediction flow of TED are as follows. Word embedding will be used to learn the interrelationship between apps in the app usage order

information. After that, to understand the time relationship between apps, the embedded results will be converted into two-dimensional metrics with time-series data encoding algorithms and then learned into a deep learning model. The deep learning model for TED will use a CNN-based model capable of parallel learning.

In this study, performance experiments and user validation experiments were conducted. The TED model recorded 84.5% based on Recall@5 in the Liveab dataset in the performance experiment, about 5% point higher than the previous paper using the same dataset. In addition, only about 20% of parameters were used compared to previous studies, and it was confirmed that the speed was 1 sec faster. It was also confirmed through the user validation experiment that the model training method suitable for solving the NAMLU problem is a transfer learning method.

This study reaffirmed the importance of the user's application usage history and time information. Using the TED model confirmed that the convolution could be used in the NAMLU problem. Considering the model's performance, size, and inference speed, it could be a reasonable choice. In addition, the significance of the study could be found in finding a training method suitable for the NAMLU problem through user validation experiments.

Keywords: predict next application most likely used, word embedding, time-series data encoding, Mobile, Prediction Model

Student Number: 2020 - 29879