학생 논문

GRU모델과 교통류 이론을 결합한 교통 예측 모델의 개발에 관한 연구

Nuts

최 우 진

2024년 8월

GRU모델과 교통류 이론을 결합한 교통 예측 모델의 개발에 관한 연구

최 우 진

초록

국문 학위논문의 초록은 국문, 영문의 순서로 작성하며, 영문 학위논문의 초록은 영문, 국문의 순서로 작성하며, 학위논문을 기타 외국어로 작성하는 경우 초록은 기타 외국어, 영문, 국문의 순서로 작성한다. 초록에는 논문제목, 성명, 학과, 지도교수를 기재하며 초록 하단에 주요어(keywords)를 표기한다. 페이지 번호는 초록부터 본문 전까지 작은 로마 숫자(Roman numerals, e.g., i, ii, iii, iv...)를 사용한다.

중심어 : 인공지능, 딥러닝, 빅데이터, 교통류, 교통류 분석, 교통류 예측, 신경망, 심층 신경망, 인공 신경망, 순환 신경망, GRU, 게이트 순환 유닛, 머신러닝, 장단기 기억 신경망, 시계열 데이터, 모델 최적화, 연속 학습, 성능 최적화, 데이터 전처리, 교통 관리 시스템, 예측 모델, 데이터 통합, 모델 평가, 실시간 예측, 데이터 융합, 피드백 함수, 교통 네트워크

A Study on the Development of a Traffic Prediction Model: Combining GRU and Traffic Flow Theory

by Woojin Choi

Abstract

The text of the abstract begins here.

Keywords: Artificial Intelligence, Deep Learning, Big Data, Traffic Flow, Traffic Flow Analysis, Traffic Flow Prediction, Neural Network, Deep Neural Network, Artificial Neural Network, Recurrent Neural Network, GRU, Gated Recurrent Unit, Machine Learning, Long Short-Term Memory, Time Series Data, Model Optimization, Continual Learning, Performance Optimization, Data Preprocessing, Traffic Management System, Prediction Model, Data Integration, Model Evaluation, ARIMA, Real-Time Prediction, Data Fusion, Feedback Function, Traffic Network

목차

표 목차

그림 목차

1장. 서론

1.1 연구의 배경 및 목적

대한민국의 교통 산업은 1906년 최초의 치도 계획이 수립된 이후 현재까지 지속적으로 성장하였다. 1970년 경부고속도로를 기점으로 현재 대한민국에는 총 51개의 고속도로 노선이 존재한다. 2023년 대한민국의 고속도로 총연장은 4,973km이고, 일반 국도 등을 모두 포함한 총 도로연장은 115,878km에 이른다 [?]. 이 과정에서 국민들의 도로 이용률과 전반적인 활동 범위가 크게 늘어났다. 2023년 대한민국에는 25949대의 자동차가 등록되어 있고, 평균 일교통량은 16051대까지 증가하였다 [?].

그러나 최근에는 자동차의 보급률과 교통량의 증가로 인한 다양한 문제들이 보고되고 있다. 특히 인구의 절반이 수도권에 밀집하여 거주하는 인구분포와 지나친 성장 위주의 교통 정책으로 인한 소통 혼잡과 교통체증등의 사회적, 경제적 문제가 크게 대두되고 있다. 수도권에서는 출퇴근 시간대의 교통 혼잡이 심각하며, 이는 경제적 손실뿐만 아니라 환경 오염과 교통사고의 주요 원인으로 작용하고 있다. 또한 교통체증은 내연기관 자동차 바퀴의 공회전을 유발하기에 경제적 손실뿐만 아니라, 대기오염과 에너지 소비 증가를 초래하여 환경적 문제 또한 가중시키고 있다. 도로 교통망은 국가 경제의 혈관으로서 기능하며, 그 효율성은 국가 전체의 경제활동에 중대한 영향을 미친다. 그러나 최근의 통계는 대한민국의 주요 도시들이 심각한 교통 혼잡에 직면해 있음을 보여준다. 국내 교통혼잡비용은 1994년 10조원 대비 2021

년 45.8조원으로 358% 증가하였다 [?], [?]. 이러한 문제를 해결하기 위해서는 교통 흐름을 정확히 예측하고, 효율적으로 관리할 수 있는 ITS(Intelligent Transportaion System, 지능형 교통 시스템)을 위한 고도화된 교통 예측 모델이 필수적이다.

현재 대한민국의 도로 정책은 교통 혼잡 문제를 해결하기 위한 다양한 방안을 모색하고 있으며, 교통류 예측 모델의 개발과 적용이 그 중심에 있다. 전통적인 교통류 예측 모델들은 주로 수리적 모델과 시계열 예측을 기반으로 하여 교통 흐름을 설명하고 예측하는 데 중점을 두고 있다. 수리적 분석은 Kalman Filter [?], Markov Chain [?] 등의 범용적인 방법과 LWR(Lighthill-Whitham-Richards) 모델 [?], 충격파 이론 [?] 등의 교통류 이론들로 나눌 수 있다. 시계열 예측은 대표적인 모델인 ARIMA(Autoregressive Integrated Moving average Model) [?], [?], [?] 과 계절성을 추가한 SARIMA(Seasonal ARIMA) [?] 등이 주를 이룬다. 수리적 분석은 교통류의 물리적 특성을 반영하여 교통 밀도와 속도 간의 관계를 설명하는 데 강점을 지니며, 시계열 예측은 과거의 데이터를 기반으로 미래의 교통 상황을 예측하는 데 주로 사용된다. 그러나 이러한 모델들은 교통 흐름의 비선형적 특성을 충분히 반영하지 못하며, 복잡한 교통 상황에서의 예측 정확도가 떨어지는 한계를 가지고 있다.

최근 들어 인공지능과 딥러닝 등 기계학습의 발전이 교통류 예측 연구에 새로운 가능성을 열어주고 있다. 기계학습은 Neural Network(NN) [?], [?], [?], [?] 를 중추로 하여 RNN(Recurrent Neural Network) [?], [?], [?], SVR(Support Vector Regression) [?], [?] , [?] 등의 회귀 분석 및 학습이 주를 이루고 있다 [?], [?], [?]. 딥러닝 모델, 특히 순환 신경망(RNN)과 같은 시계열 데이터를 다루는 모델들은 교통 데이터의 복잡한 패턴을 학습하고, 이를 기반으로 보다 정교한 예측을 수행할 수 있는 능력을 갖추고 있다. 이로 인해 다양한 교통 상황에서의 예측 정확도가 크게 향상되었으며, 교통 관리와 최적화에 있어 중요한 도구로 자리매김하고 있다. 그러나, 현재까지의 연구는 주로 개별 모델의 성능 향상에 초점을 맞추고 있으며, 서로 다른 모델의 장점을 결합하여 예측 성능을 극대화하려는 시도는 상대적으로 미진한

실정이다.

특히, 교통류 예측 분야에서 결합 모델에 대한 연구는 아직 초기 단계에 머물러 있다. 전통적인 교통류 모델과 딥러닝 모델을 통합하여 예측의 정확성과 신뢰성을 높이려는 시도는 일부 진행되고 있으나, 이러한 연구는 주로 이론적 가능성에 그치고 있다. 결합 모델은 각기 다른 접근법의 장점을 통합함으로써 보다 풍부한 예측 정보를 제공할 수 있는 잠재력을 지니고 있음에도 불구하고, 모델의 복잡성과 데이터 요구 사항으로 인해 실질적인 연구와 실증적 검증이 제한적으로 이루어지고 있다.

본 연구는 이러한 한계를 극복하고, 교통류 예측 모델의 새로운 방향을 제시하기 위해 LWR(Lighthill-Whitham-Richards) 모델 [?], 충격파 이론 [?], 뉴웰의 관성 모델(Newell's Car-Following Model) [?], 그리고 3상 교통 이론(Three-Phase Traffic Theory) [?], [?] 의 교통류 이론들과 GRU(Gated Recurrent Unit) 모델 [?] 을 결합한 새로운 예측 모델을 제안하고자 한다. 교통류 모델들은 차량 간의 상호작용과 흐름의 변화를 설명하는 데 효과적이며, GRU 모델은 시계열 데이터의 장기 의존성을 학습하는 데 탁월한 성능을 보인다. 두 모델의 결합은 교통 흐름의 미시적 특성과 거시적 패턴을 동시에 반영할 수 있는 강력한 예측 도구를 제공할 것으로 기대된다. 이에 따라, 본 연구의 주요 목적은 교통류 이론과 GRU 모델을 결합하여 교통 흐름의 예측 정확도를 향상시키는 것이다.

1.2 연구의 범위 및 방법

1.2.1 연구의 범위

1) 공간적 범위

결합 모델 및 타 비교 모델들의 학습에 사용된 데이터는 국가교통정보센터의 표준노드링크 위치 데이터에 기반하여 병합되었다. 표준노드링크 데이터는 속도변화

발생점인 노드와 노드를 연결한 선을 의미하는 링크와[1], 차량이 도로를 주행함에 있어 속도의 변화가 발생되는 곳을 표현하는 노드로 구성되어 있으며, 이에 따라 더욱 자세한 도로 정보를 학습할 수 있다 [?]. 노드 데이터는 전국 총 231716개 존재하며, 위치 데이터의 범위는 대한민국 대다수의 주요 도로를 포함하고 있다 ??[2]. 노드링크 데이터는 다양한 도로 특성을 가지고 있으며, 고속도로, 국도, 지방도로, 그리고 도시 내 도로를 포함한 다양한 유형의 도로로 구성되어 있다. 표준노드링크 데이터는 2024년 3월 25일 제공된 데이터를 수집하여 사용하였다.

Figure 1.1: 표준노드링크 데이터 시각화

[1]즉, 실세계에서의 도로이다.
[2]shp 파일 시각화 프로그램으로 QGIS를 사용하였다.

2) 시간적 범위

교통 예측을 위해서는 비선형적 특성과 다양한 교통 정보들을 분석해야 한다. 따라서 본 연구는 2023년의 1, 4, 7, 10월의 각종 교통 정보 데이터를 사용하였다. 이러한 시간적 구분은 기상 조건이 교통 흐름에 미치는 영향을 심층적으로 분석하기 위한 전략적 선택에 근거한다. 교통류의 흐름은 계절의 특성과 기상 상태에 따라 크게 좌우되므로, 본 연구에서는 각 분기 첫째 달의 데이터를 사용하므로서 기상 요인에 의한 교통류 패턴을 효율적으로 분석 및 예측 하고자 한다.

1.2.2 연구의 방법 및 내용

본 연구의 내용은 크게 관련 이론 및 기존 연구 고찰, 데이터 구성, 결합 교통 예측 모델 개발, 실험 및 성능 평가로 나뉘어져 있다. 먼저, 본 연구는 교통 예측 모델을 개발 혹은 활용한 기존 연구, 그리고 교통류 이론과 딥러닝 등 본 연구의 관련 이론을 고찰하였다. 그 이후 각종 공공 교통 및 기상 원시 데이터를 수집하여 분석하였다. 수집된 데이터는 분석 및 모델 학습에 적합한 형태로 변환하기 위해 전처리 과정을 거친다. 전처리 과정은 결측치 처리, 이상치 제거, 정규화 등의 단계를 포함하며, 유의 변수를 선정해 분석 데이터의 형태로 가공하여 병합하였다. 이를 통해 데이터의 일관성을 유지하고, 신경망이 효과적으로 학습할 수 있도록 하였다.

본 연구의 핵심은 교통류 이론과 순환 신경망을 효과적으로 개발하여 성능을 입증하는 것이다. 모델 개발 단계에서는 LWR, 충격파 이론 등의 교통류 이론 특성을 분석하고, 적합한 결합 이론을 선정한다. 또 순환 신경망은 결합 모델로서 연산량이 상대적으로 적어 교통류 이론과 결합하여도 연산속도를 보장할 수 있는 GRU로 선정하였다. GRU의 연산속도, 예측 정확도 등을 MAE, MSE, R^2 score 등의 회귀 모델 평가 지표[3]를 활용하여 검정한다. 최종적으로, 선정된 교통류 이론과 결합 모델의

[3]MAE, MSE, RMSE, R^2 score를 사용하였다.

특성을 논리적으로 고려하여 교통 예측 결합 모델을 개발하고자 한다.

마지막으로, 개발된 교통 예측 결합 모델과 성능을 비교할 기존의 교통 예측 모델을 프로그래밍 언어 Python을 사용하여 구현한다. 구현된 예측 모델들은 거시적, 미시적 범위로 면밀히 비교되고, 개발된 결합 모델에 종합적인 평가를 내린다.

마지막으로, Case Study에서는 인구가 가장 밀집한 서울 지역을 대상지로 선정하여 적합한 모델 평가와 교통 분석을 진행하고자 한다. 이를 통해 더욱 대한민국에 적합한, 또 실재적인 교통류 예측이 가능할 것이다.

추가적으로, 모든 모델에 공통적으로 연속 학습(Continual Learning)을 적용시켜 모델의 범용성과 데이터 일반화 능력을 향상시켰고, 모델의 예측을 자기 회귀적 예측법과 외부 요인 기반 예측법으로 구분하여 시행해 모델의 예측 정확도를 더욱 면밀하게 관측할 수 있게 하고자 한다.

본 과정을 통해 본 논문의 궁극적 목표인 교통류 이론과 GRU모델을 결합한 개량 교통 예측 모델을 개발하고자 한다 **??**.

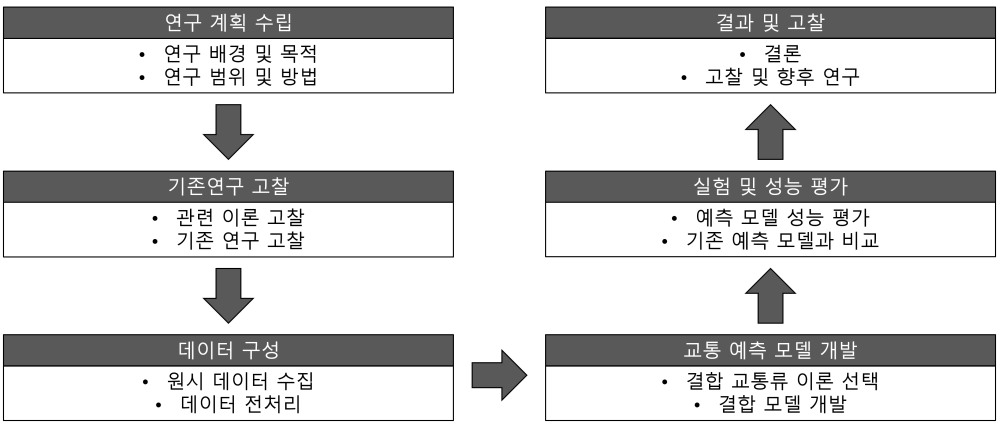

Figure 1.2: 연구의 수행 절차

2장. 이론 및 선행연구 고찰

2.1 관련 이론 고찰

2.1.1 인공 신경망(ANN)

1) 퍼셉트론

퍼셉트론은 프랭크 로젠블랫(Frank Rosenblatt)이 1958년에 제안한 초기 인공신경망 알고리즘이다 [**?**]. 퍼셉트론은 단층 신경망으로서, 선형 분리 가능한 데이터를 분류하는 데 사용된다. 입, 출력 x_i과 $\hat{y}$은 숫자 값이고 각 입력에 고유 가중치 W_i을 곱하는 방식으로 분류가 진행된다. 곱해진 값은 모두 합산되어, 활성화 함수인 계단 함수에 전달되어, 결과 $\hat{y}$를 출력하게 된다 **??**.

마빈 민스키(Marvin Minsky)와 시모어 페퍼트(Seymour Papert)가 1969년 단층 퍼셉트론은 베타적 논리합 회로(XOR)조차 해결하지 못한다는 문제를 찾아내었다 [**?**]. 또한 퍼셉트론 단층 구조의 한계를 지적하며 다층 구조 퍼셉트론을 제안하였다.

$$\hat{y} = f\left(\sum_{i=1}^{n} W_i x_i\right) \quad f(a) = \begin{cases} 0 & \text{if } a < \beta \\ 1 & \text{if } a \geq \beta \end{cases}$$

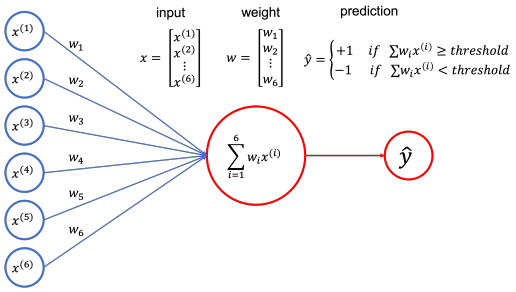

Figure 2.3: 퍼셉트론(Perceptron)

2) 다층 퍼셉트론

다층 퍼셉트론(Multilayer Perceptron, MLP)은 인공신경망(Artificial Neural Network, ANN)의 기본적인 구조 중 하나로, 비선형적으로 분리 불가능한 문제를 해결하기 위해 퍼셉트론(Perceptron)의 개념을 확장한 모델이다. 다층 퍼셉트론은 입력층(input layer), 하나 이상의 은닉층(hidden layer)[4], 그리고 출력층(output layer)으로 구성된다. 각 층은 여러 개의 노드(node)로 이루어져 있으며, 이러한 노드들은 가중치(weight)를 통해 서로 연결되어 있다. 입력층은 외부 데이터의 피처(feature)를 받아들이고, 은닉층은 이 데이터를 비선형적으로 변환하며, 출력층은 최종 예측값을 출력하는 역할을 한다.

다층 퍼셉트론과 단층 퍼셉트론의 주요 차이점 중 하나는 은닉층에 활성화 함수(activation function)를 사용하여 비선형성을 도입하는 것이다. 활성화 함수는 신경망이 단순히 선형 분류 가능 데이터만을 학습하는 것이 아닌, 비선형적인 복잡한 데이터를 분류하게 해준다. 일반적으로 사용되는 활성화 함수로는 시그모이드(sigmoid) 함수, 하이퍼볼릭 탄젠트(hyperbolic tangent, tanh) 함수, 그리고 ReLU(Rectified

[4]은닉층이 2개 이상인 신경망을 심층 신경망(Deep Neural Network, DNN)이라고 한다.

Linear Unit) 함수 등이 있다.

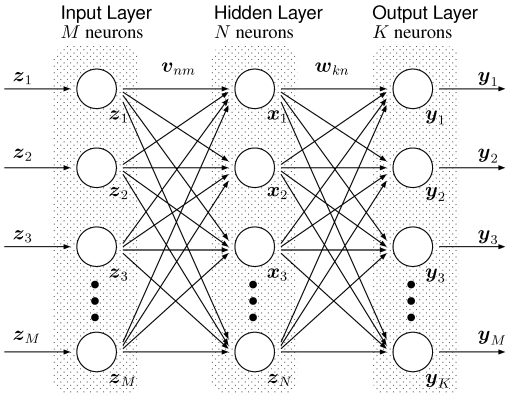

Figure 2.4: 다층 퍼셉트론(Multilayer Perceptron)

3) 활성화 함수

활성화 함수란 입력값을 받아 특정 연산을 수행한 후, 출력값을 통해 그 신호가 다음 단계에서 활성화되는지를 결정하는 함수이다. 신경망에서 활성화 함수는 매우 중대한 역할을 하며, 출력된 값이 다음 층의 뉴런에 전달되는지를 결정한다. 초기에는 퍼셉트론과 같은 단순한 신경망에서 0과 1의 값만을 출력하는 계단 함수(step function)가 사용되었으나, 정보가 축소되어 손실된다는 단점으로 현재는 거의 사용되지 않는다. 이를 개선하기 위해, 0과 1 사이의 연속적인 값을 출력하는 시그모이드(Sigmoid) 함수가 일반적으로 많이 사용되었다. 시그모이드 함수는 각 뉴런의 출력값을 0과 1 사이로 매핑하여, 뉴런이 활성화될 확률을 출력한다. 그러나 시그모이드 함수는 입력 값이 크거나 작을 때 기울기가 0에 가까워지는 기울기 소실(Vanishing Gradient) 문제를 일으킬 수 있다는 단점이 있다. 이와 같은 문제를 해결하기 위해 많이 사용되는 활성화 함수 중 하나가 ReLU(Rectified Linear Unit) 함수이다. ReLU 함수는 입력이 0보다 클 때는 입력 값을 그대로 출력하고, 0 이하일 때는 0을 출력하

는 함수이다 **??**.

$$\mathrm{ReLU}(x) = \max(0, x)$$

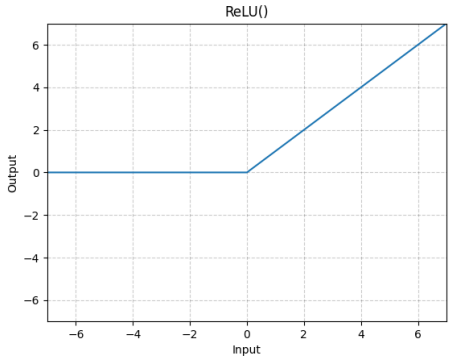

Figure 2.5: ReLU(Rectified Linear Unit)

2.1.2 신경망 학습

인공 신경망은 학습 과정에서 여러 가지 문제가 발생할 수 있다. 이러한 문제들은 신경망의 성능을 저하시킬 수 있으며, 모델이 효과적으로 학습하지 못하도록 방해한다.

기울기 소실 및 폭주

기울기 소실 문제는 신경망의 학습 과정에서 역전파 알고리즘을 사용하는 중에 발생한다. 이 문제는 네트워크의 층이 깊어질수록 손실 함수의 기울기가 점점 작아져, 초기 층의 가중치가 거의 업데이트되지 않게 되는 현상을 말한다. 이는 신경망이 충분히 학습되지 않게 하여, 인공 신경망이 데이터의 패턴을 제대로 학습하지 못하게

만든다. 기울기 폭주 문제는 역전파 과정에서 기울기가 지나치게 커져 가중치가 불안정하게 증가하는 현상을 의미한다. 기울기 폭주 문제는 주로 깊은 신경망이나 재귀 신경망에서 나타나며, 인공 신경망의 학습이 수렴하지 않고 발산하게 만들어 성능을 저하시킬 수 있다.

과적합

과적합은 신경망이 훈련 데이터에 지나치게 맞춰져, 새로운 데이터나 테스트 데이터에서 성능이 저하되는 현상을 의미한다. 이는 모델이 훈련 데이터의 잡음까지 학습하여, 일반화 능력이 떨어지게 되는 것이다. 과적합된 모델은 실제 데이터에서 좋은 성능을 보이지 못하며, 이는 신경망이 새로운 데이터를 학습하는 데 어려움을 겪게 만든다.

데이터 편향 및 불균형

데이터가 편향되어 있거나 불균형할 경우, 신경망은 특정 패턴이나 클래스에 대해 편향된 학습을 하게 된다. 이는 예측 모델의 성능을 왜곡시키고, 특정 클래스나 패턴을 과소, 과대평가하게 만들어 신경망의 학습 결과에 부정적인 영향을 미친다.

1) 옵티마이저

옵티마이저(Optimizer)는 신경망 학습에서 모델의 가중치를 업데이트하는 알고리즘으로, 손실 함수(Loss Function)의 값을 최소화하여 모델의 성능을 향상시키는 데 중요한 역할을 한다. 신경망은 주어진 입력 데이터에 대한 예측 값을 출력하고, 그 예측이 실제 값과 얼마나 차이가 있는지를 손실 함수를 통해 평가한다. 옵티마이저는 이 손실 값을 최소화하기 위해 모델의 가중치 매개변수(parameter)를 조정하여, 예측 성능을 향상시킨다.

옵티마이저는 손실 함수의 현재 지점 기울기(Gradient)를 계산하여, 손실 함수의 값이 줄어들 수 있는 방향으로 전역 최소점(Global Minimum)을 찾아 모델의 매개변수를 업데이트하는 것이다[5]. 기울기는 손실 함수의 값이 어떻게 변하는지를 나타내는 지표로, 편미분을 통해 기울기가 큰 방향으로 매개변수를 이동시키면 손실 값이 감소하게 된다. 이 과정은 경사 하강법(Gradient Descent)을 기반으로 하며, 신경망의 학습 과정에서 반복적으로 수행된다 **??**. 경사 하강법은 함수의 기울기를 구하여 기울기가 낮은 쪽으 로 계속 이동시켜 미분값이 0이 되는 지점까지 반복하는 옵티마이징 방법이다. 최적화할 함수 $f(x)$에서 시작점 x_0 를 정하고, x_i가 주어지면, 다음 이동점인 $x_i + 1$가 산출된다. 이때 γ_i는 한번의 연산에서 이동할 폭을 조절하는 하이퍼 파라미터로 학습률(learning rate)라 한다. 학습률을 크게 하면 발산되고 작게 하면 수렴이 되지 않아 전역 최소점 탐색에 차질이 생기기 때문에 적절한 값을 설정해야 한다.

$$x_{i+1} = x_i - \gamma_i \nabla f(x_i)$$

가. 모멘텀 옵티마이저(Momentum Optimizer)

보리스 폴락(Boris T. Polyak)이 1964년 발표한 옵티마이저인 모멘텀 옵티마이저는 기존 경사 하강법에 실세계의 물리 법칙을 결합시킨 방법이다. 모멘텀 옵티마이저는 느리게 출발하여 특정 시점부터 가속되는 관성의 원리가 적용되어 있다. 모멘텀 옵티마이저는 이전 기울기(Gradient)의 방향과 크기를 일정 부분 유지하면서 새로운 기울기를 반영하여, 더욱 빠르고 안정적인 수렴을 도모한다.

모멘텀 옵티마이저는 각 업데이트 단계에서 이전 단계의 이동 방향을 일정 부분 유지하는 방식을 채택한다. 이를 통해, 기울기가 지속적으로 동일한 방향으로 작용

[5]전역 최소점(Global Minimum)을 찾는 과정에서 함수가 비선형성을 띌 경우, 신경망이 지역 극소점(Local Minima)에 빠질 수 있다.

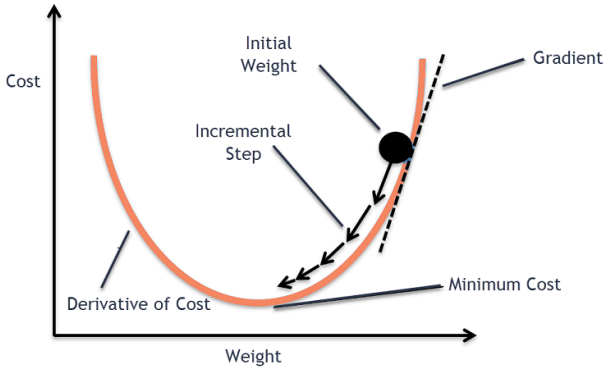

Figure 2.6: 경사 하강법(Gradient Descent)

하는 경우에는 업데이트 속도를 가속화하고, 반대로 기울기가 자주 변하는 경우에는 진동을 줄여 안정적으로 수렴할 수 있게 한다.

$$v_{t+1} = \beta v_t + (1-\beta)\nabla f(x_t)$$

$$x_{t+1} = x_t - \eta v_{t+1}$$

여기서 v_t는 시간 t에서의 이동 속도(velocity)를 나타내며, β는 모멘텀 계수(momentum coefficient)로, 일반적으로 0과 1 사이의 값을 갖는다. $\nabla f(x_t)$는 현재 지점 x_t에서의 손실 함수 $f(x)$의 기울기이며, η는 학습률(learning rate)이다. 모멘텀 계수 β는 이전 이동 속도를 얼마나 반영할지를 결정하는 파라미터로, 값이 클수록 관성이 크게 작용하여 이동 방향이 더 강하게 설정된다.

나. AdaGrad

AdaGrad(Adaptive Gradient Algorithm)는 기계 학습과 딥러닝 모델의 최적화를 위한 경사하강법(Gradient Descent) 기반의 옵티마이저 중 하나로, 2011년 John

Duchi, Elad Hazan, Yoram Singer에 의해 제안되었다. AdaGrad는 학습률(Learning Rate)을 각 매개변수에 대해 각 매개변수의 과거 기울기 제곱합의 역수에 비례하여 학습률을 조정합니다. 이를 통해, 자주 업데이트되는 매개변수의 학습률은 감소시키고, 드물게 업데이트되는 매개변수의 학습률은 증가시킨다.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \nabla f(\theta_t)$$

$$G_t = \sum_{i=1}^{t} \nabla f(\theta_i)^2$$

- θ_t는 시간 t에서의 매개변수
- η는 초기 학습률
- G_t는 시간 t까지의 기울기 제곱합
- $\nabla f(\theta_t)$는 현재 지점에서의 손실 함수의 기울기
- ϵ은 0으로 나누는 것을 방지하기 위한 작은 값

다. RMSProp

AdaGrad의 단점을 해결하기 위해 제프리 힌튼이 제안한 알고리즘이다. 아래의 식과 같이 AdaGrad의 기울기의 제곱값을 더해 구한 G_t 를 지수평균으로 사용한다.

$$G = \gamma G + (1 - \gamma)(\nabla_\theta J(\theta))^2$$

$$\theta = \theta - \frac{\eta}{\sqrt{G + \epsilon}} \nabla_\theta J(\theta)$$

라. Adam

Adam(Adaptive Moment Estimation) 옵티마이저는 2014년 Diederik P. Kingma와 Jimmy Lei Ba에 의해 제안되었다. Adam은 학습 과정에서 각 매개변수의 1차 모멘트(기울기의 평균)와 2차 모멘트(기울기의 제곱 평균)를 동시에 고려하여, 학습률을 적응적으로 조정한다.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla f(\theta_t)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\nabla f(\theta_t))^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}}\hat{m}_t$$

- m_t는 시간 t에서의 1차 모멘트(기울기의 지수 이동 평균)
- v_t는 시간 t에서의 2차 모멘트(기울기 제곱의 지수 이동 평균)
- β_1와 β_2는 각각 1차와 2차 모멘트의 지수 이동 평균을 계산할 때 사용되는 감쇠 계수(decay rate)
- $\hat{m}_t$와 $\hat{v}_t$는 편향을 보정한 1차와 2차 모멘트 추정치

- η는 학습률(learning rate)
- ϵ은 수치적 안정성을 위해 사용되는 작은 상수이다

2) 오차 역전파

오차 역전파(Backpropagation)은 1986년 데이비드 E. 루멜하트(David E. Rumelhart), 제프리 힌튼(Geoffrey Hinton), 로널드 J. 윌리엄스(Ronald J. Williams)에 의해 체계적으로 정립되었다. 이 알고리즘은 다층 퍼셉트론(Multilayer Perceptron, MLP)과 같은 심층 신경망(Deep Neural Networks)의 학습에서 사용되며, 네트워크의 출력을 실제 값에 근사시키기 위해 가중치(weight)와 편향(bias)을 조정한다. 오차 역전파 알고리즘은 네트워크의 출력층에서 계산된 오차를 기준으로, 오차를 네트워크의 각 층을 거슬러 올라가며 전파함으로써 각 가중치에 대한 손실 함수(Loss Function)의 기울기를 계산한다. 이를 통해, 신경망은 출력값과 실제 목표값 간의 오차를 최소화하기 위해 각 층의 가중치를 업데이트하게 된다.

오차 역전파 알고리즘은 두 단계로 나뉜다. 첫 번째 단계는 순전파 단계(Forward Pass)로, 입력 데이터를 신경망에 통과시키면서 각 층의 유닛을 활성화하고 최종 출력층에 도달하여 예측값을 생성한다. 이 과정에서 각 유닛은 활성화 함수(Activation Function)를 통해 비선형 변환을 수행한다. 두 번째 단계는 역전파 단계(Backward Pass)로, 출력층에서 계산된 오차를 바탕으로 네트워크의 가중치에 대한 손실 함수의 기울기를 계산한다. 이때 체인 룰(chain rule)을 사용하여, 출력층에서 시작하여 입력층 방향으로 오차를 전파하면서 각 가중치의 기울기를 계산한다.

$$\delta_j^l = (\sum_k \delta_k^{l+1} w_{jk}^{l+1}) f\prime(z_j^l)$$

여기서 δ_j^l는 층 l에 있는 뉴런 j의 오차, w_{jk}^{l+1}는 층 $l+1$의 뉴런 k로의 가중치, $f\prime(z_j^l)$는 뉴런 j의 활성화 함수의 도함수이다. 이를 이용해 각 가중치 w_{ij}^l에 대한 손실

함수의 기울기 $\frac{\partial L}{\partial w_{ij}^l}$를 구할 수 있다.

$$w_{ij}^l = w_{ij}^l - \eta \frac{\partial L}{\partial w_{ij}^l}$$

5) 드롭아웃

드롭아웃은 2012년 제프리 힌튼에 의해 제안되었다. 매 학습에서 각 노드는 임시적으로 드롭아웃될 확률 p를 가지게 되며, 노드 간의 연결을 임의로 삭제할 수 있다. 드롭아웃은 신경망이 데이터의 잡음이나 불규칙성에 덜 민감하게 반응하도록 하여, 테스트 데이터에서의 성능을 개선한다 **??**.

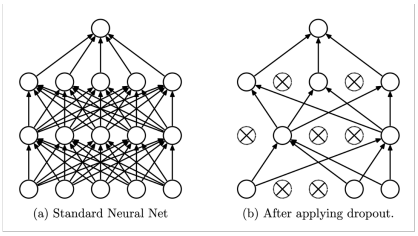

Figure 2.7: 드롭아웃(Dropout)

2.1.3 순환신경망(RNN)

1) 완전 순환 신경망

완전 순환 신경망(Fully Recurrent Neural Network, FRNN)은 순환 신경망(Recurrent Neural Network, RNN) 구조 중 하나로, 모든 뉴런이 다른 모든 뉴런과 연결되어 있다. 이 구조는 각 뉴런이 자신을 포함한 다른 모든 뉴런으로부터 입력을 받도록 하여, 내부의 모든 정보가 시간에 따라 지속적으로 순환한다. 완전 순환

신경망은 시계열 데이터 같은 시간에 따른 패턴을 학습하는 데 적합한 모델로, 특히 시퀀스 데이터(sequence data)에서의 복잡한 상호 의존성을 효과적으로 포착한다.

$$h_t = f(W \cdot h_{t-1} + U \cdot x_t + b)$$

- h_t는 시간 t에서의 은닉 상태(hidden state)
- W는 은닉 상태 간의 가중치 행렬(weight matrix)
- U는 입력 x_t에 대한 가중치 행렬
- x_t는 시간 t에서의 입력 벡터
- b는 편향 벡터(bias vector)
- f는 비선형 활성화 함수(activation function)

4) LSTM

LSTM(Long Short-Term Memory) 장기 의존성(Long-Term Dependencies)을 효과적으로 학습하기 위해 1997년 호흐라이터(Sepp Hochreiter)와 슈미드후버(Jürgen Schmidhuber)에 의해 처음 제안되었다. LSTM 네트워크는 정보의 흐름을 조절하는 여러 개의 게이트(gates)를 통해 데이터를 처리하여, RNN의 기울기 소실(Vanishing Gradient) 문제를 해결하였다. 기본적인 LSTM 셀은 입력 게이트(Input Gate), 망각 게이트(Forget Gate), 출력 게이트(Output Gate)로 구성되어 있으며, 이 게이트들은 정보가 셀 상태(Cell State)를 통해 전달되는 방식을 통제한다.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}t = \tanh(W_C \cdot [ht - 1, x_t] + b_C)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$

- x_t는 시간 t에서의 입력 벡터
- h_t는 시간 t에서의 은닉 상태
- C_t는 시간 t에서의 셀 상태
- W_f, W_i, W_C, W_o는 각 게이트에 대한 가중치 행렬
- b_f, b_i, b_C, b_o는 각 게이트의 편향 벡터
- σ는 시그모이드 활성화 함수
- tanh는 하이퍼볼릭 탄젠트 활성화 함수
- $\odot$는 원소별 곱(element-wise multiplication)

5) GRU

GRU(Gated Recurrent Unit)은 2014년 조경현(KyungHyun Cho)이 제안한 모델이다. GRU는 장단기 메모리(Long Short-Term Memory, LSTM) 네트워크의 성능을 유지하면서도 구조를 단순화하여 계산 효율성을 높였다. GRU는 LSTM과 마찬가지로 기울기 소실(Vanishing Gradient) 문제를 완화하고, 장기 의존성을 효과적으로 학습한다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$$

$$\tilde{h}t = \tanh(W_h \cdot [r_t \odot ht - 1, x_t] + b_h)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

2.2 선행연구 고찰

2.2.1 딥러닝 기반 선행연구

Nicholas G. Polson, Vadim O. Sokolov(2017)은 l_1 정규화를 사용하여 학습한 선형 교통류 예측 모델과 tanh층을 결합한 모델 아키텍처를 개발하고, 성능 검정을 완료하였다. 주된 기여는 선형 모델과 비선형적인 활성화 함수를 결합한 딥러닝 아키텍처를 제안한 것이다. 기존의 교통 흐름 예측 문제는 자유 흐름(free flow), 정체 발생(breakdown), 회복(recovery), 혼잡(congestion) 사이의 급격한 비선형적 전환으로 인해 예측이 복잡해지는데, 딥러닝 아키텍처는 이러한 시공간적 비선형 효과를 효과적으로 포착할 수 있었다. 첫 번째 층에서는 예측 변인 간의 시공간적 관계를 식별하며, 이후의 층에서는 비선형적인 관계를 모델링하였다. 연구는 미국 시카고의

고속도로(I-55)에서 수집된 교통 센서 데이터를 활용하여, 시카고 베어스 풋볼 경기와 폭설 사건과 같은 두 가지 특수 이벤트 동안의 교통 흐름을 예측하였다. 연구의 Case Study 결과는 다음과 같았다. **??**.

	DLL	DLM8L	DLM8	DLTF15L	DLTF15	VARM8L	VARTF15L
IS MSE	13.58	7.7	10.62	12.55	12.59	8.47	15
IS R^2	0.72	0.83	0.76	0.75	0.75	0.81	0.7
OS MSE	13.9	8.0	9. 5	11.17	12.34	8.78	15.35
OS R^2	0.75	0.85	0.82	0.81	0.79	0.83	0.74

Table 2.1: Interstate I-55 Case Study 결과

B. Gültekin Çetiner, Murat Sari, Oğuz Borat(2010)은 이스탄불의 이스탄불 교통 유지 관리 부서(ISBAK)에서 제공된 도로 교통 데이터를 기반으로 기본 인공신경망(ANN)을 사용하여 예측 분석을 진행하였다. 연구에서는 ANN의 백프로파게이션 알고리즘을 사용하여 두 가지 시간 간격(5분 및 1시간)에 따른 예측 성능을 비교하였다. 130,701개의 값으로 구성된 데이터셋을 사용하고 학습되었고, 상관 계수 0.93[6]과, 0.85[7]이라는 고무적인 결과를 내었다.

Hongsuk Yi, HeeJin Jung(2017)은 Google의 TensorFlow™기반의 DNN(Deep Neural Network)을 사용하여 교통 혼잡도를 예측하였다. 기존의 교통 예측 모델들은 인공신경망(ANN) 기반의 좁은 네트워크 구조와 제한적인 학습 샘플로 인해 부분적인 성공에 그쳤다. 이에 이 연구에서는 Google의 TensorFlow™프레임워크를 사용한 심층신경망(DNN)을 도입하여, 교통 흐름의 정체 상태와 비정체 상태를 구분하고 예측할 수 있는 새로운 모델을 제시하였다. 제안된 모델은 실시간 데이터를 5분 간격으로 수집하여 학습하며, 로지스틱 회귀 분석을 통해 교통 성능 지수(TPI)를 계산하여 교통 흐름의 상태를 구분한다. 이 모델은 3개의 은닉층으로 구성되었으며,

[6]훈련 세트의 처음 20,000개 데이터로 훈련되었으며, 테스트는 나머지 테스트 세트를 통해 수행된 결과

[7]각각 1-1,000 및 1,001-2,015의 범위로 훈련 및 테스트 세트로 나뉘어서 수행된 결과

하이퍼볼릭 탄젠트 함수(tanh)를 활성화 함수로 사용하여 신경망을 훈련한다. 제안된 TensorFlow™DNN 모델은 3개의 레이어로 구성되었으며, 각각 40, 50, 40개의 뉴런을 포함했다. 다만 메모리 용량 및 CPU 계산 시간의 제한으로 인해 전체 데이터 중 약 1%만을 훈련에 사용할 수 있었으며, 이를 해결하기 위해 향후 GPU를 활용한 고성능 플랫폼을 구축하여 모델을 개선할 필요가 있다. 이 모델은 하이퍼탄젠트 함수(tanh)를 활성화 함수로 사용하며, 약 99%의 정확도로 교통 흐름 상태를 예측할 수 있음을 보였다.

Guo Xiaojian, Zhu Quan(2009)은 BP 신경망(Backpropagation Neural Network)을 사용하여 2008년 6월 20일부터 7월 25일까지 중국 간저우시에 위치한 홍치 교차로의 역사적 데이터를 학습시켰다. 연구에서는 중국 간저우 시의 홍치 교차로에서 수집된 실시간 교통 데이터를 기반으로 MATLAB을 통해 BP 신경망 모델을 구현하였다. 실험 결과, BP 신경망 모델은 교차로 교통 흐름 예측에서 신호 주기와 교차로의 교통 흐름 변동성을 효과적으로 반영하여 예측 성능을 향상시켰다. 특히, Levenberg-Marquardt 알고리즘을 사용하여 BP 신경망의 학습 과정을 최적화하였으며, 학습 시간이 증가할수록 예측 정확도가 점진적으로 향상되는 경향을 보였다. 제안된 모델은 교차로의 교통 흐름 예측 정확도 약 85%를 보여주었으며, 이는 도시 교차로에서 단기 교통 흐름 예측을 위한 새로운 신뢰할 수 있는 방법으로 딥러닝이 작용할 수 있음을 시사한다.

Danqing Kang, Yisheng Lv, Yuan-yuan Chen(2017)은 LSTM(Long Short Term Memory)신경망을 사용하여 속도, 점유율, 인접 도로의 교통 정보 등으로 구성되어 있는 교통 데이터를 학습시켰다. 연구에서는 16개의 서로 다른 LSTM 모델을 사용하여 단기 교통 흐름 예측 성능을 평가하였으며, 이 모델들은 입력 변수와 네트워크 구성을 달리하여 다양한 실험 조건 하에서 비교되었다. 입력 변수로는 속도(speed), 도로 점유율(occupancy), 이웃 교통 정보가 사용되었으며, 단일 변수를 사용하는 모델과 두 가지 혹은 세 가지 변수를 조합한 다변량 모델들이 각각 평가되었다. 특히,

이웃 교통 정보를 추가로 사용하는 모델들은 특정 도로의 교통 흐름이 인접 도로의 교통 상황에 의해 영향을 받을 수 있음을 반영하여, 공간적 상관성을 고려한 예측 성능을 분석하였다. 다변량 데이터를 입력으로 사용한 모델이 단일 변수를 입력으로 사용한 모델에 비해 전반적으로 더 나은 성능을 보였으며, 특히 속도, 도로 점유율, 이웃 교통 정보를 모두 결합한 모델이 가장 높은 예측 정확도를 기록하였다 **??**.

Model	Number of input variables	Detector station for prediction							RMSE	MAE	MAPE (%)	Sorting
		U3	U2	U1	P	D1	D2	D3				
1	1				*				23.61	16.16	13.22	16
2	2			*	*				23.45	16.37	13.15	15
3					*	*			18.82	13.73	10.50	8
4	3		*	*	*				23.34	16.28	13.05	14
5				*	*	*			18.54	13.51	10.41	6
6					*	*	*		21.99	15.87	11.42	12
7	4	*	*	*	*				23.17	16.17	13.04	13
8			*	*	*	*			18.49	13.43	10.29	5
9				*	*	*	*		21.96	15.85	11.22	11
10					*	*	*	*	18.78	14.02	10.17	7
11	5	*	*	*	*	*			17.77	12.89	9.97	3
12			*	*	*	*	*		21.92	15.76	11.20	10
13				*	*	*	*	*	18.32	13.56	9.96	4
14	6	*	*	*	*	*	*		21.27	15.26	10.91	9
15			*	*	*	*	*	*	18.26	13.55	9.94	2
16	7	*	*	*	*	*	*	*	17.18	12.79	9.26	1

Table 2.2: Performance of 16 LSTM Models with Neighbor Flow as Inputs(2017)

Yongxue Tian, Li Pan(2015)은 LSTM을 활용하여 기존 교통 예측 방법론들과의 성능을 비교하였다. 이 연구에서 사용된 데이터는 Caltrans Performance Measurement System(PeMS)으로부터 수집되었으며, 캘리포니아 전역의 고속도로 시스템에서 매 30초마다 수집된 교통 흐름 데이터를 5분 간격으로 집계한 것이다. 이러한 데이터를 바탕으로, LSTM RNN 모델의 성능을 다양한 예측 간격(15분, 30분, 45

분, 60분)에 대해 평가하였다. 또한, LSTM 모델의 성능을 검증하기 위해 랜덤 워크(RW), 서포트 벡터 머신(SVM), 단층 피드포워드 신경망(FFNN), 스택드 오토인코더(SAE)와 비교하였다. LSTM 모델은 훈련 과정에서 BPTT(Backpropagation Through Time) 알고리즘을 사용하여, 장기적인 시계열 데이터를 효과적으로 학습하였다. LSTM은 다른 모델들과 비교하여 MAPE(Mean Absolute Percentage Error)와 RMSE(Root Mean Square Error)에서 가장 낮은 값을 기록하며, 더 높은 예측 정확도를 보였다. 특히, 15분 간격 예측 실험에서는 LSTM이 6.49%의 MAPE와 50.94의 RMSE를 기록하여, 6.58%의 MAPE와 53.10의 RMSE를 기록한 SVM 모델보다 성능이 우수함을 입증하였다 **??**. LSTM은 입력 데이터의 길이가 짧아도 뛰어난 예측 성능을 유지하며, 긴 입력 데이터를 학습할 때에도 적절한 시간 지연을 동적으로 결정하여 메모리 능력을 발휘하였다. 또한, 실험에서 LSTM 모델은 입력 데이터의 히든 유닛 수에 따른 성능 변화를 분석한 결과, 히든 유닛 수가 20을 초과했을 때 성능이 안정화되는 경향을 보였다. 이는 모델의 복잡성이 너무 낮을 때는 예측 성능이 저조하지만, 지나치게 복잡해지면 성능 향상이 둔화되거나 오히려 악화될 수 있음을 시사한다.

Models	15-min		30-min		45-min		60-min	
	MAPE(%)	RMSE	MAPE(%)	RMSE	MAPE(%)	RMSE	MAPE(%)	RMSE
RW	10.39	81.79	14.57	217.32	20.70	435.92	26.86	731.51
SVM	6.58	53.10	6.16	102.12	6.98	175.35	7.36	252.21
FFNN	9.72	63.38	11.65	122.77	11.82	203.36	12.98	285.84
SAE	7.63	55.84	7.61	109.66	8.58	179.39	8.97	254.57
LSTM RNN	**6.49**	**50.94**	**5.80**	**95.80**	**6.37**	**159.44**	**6.25**	**233.01**

Table 2.3: Performance of 16 LSTM Models with Neighbor Flow as Inputs(2017)

Fu Rui, Zhang Zuo, Li Li(2016)은 LSTM과 GRU를 이용하여 설계된 교통 예측 모델과 기존 방법론들인 ARIMA 등과 비교를 진행하였다. 평가 지표는 MSE, MAE

를 사용하였으며, LSTM과 GRU 모두 ARIMA를 능가하는 예측 정확도를 보여주었다 **??**. LSTM은 더 복잡한 데이터셋에서도 안정적인 성능을 보였으며, 특히 장기적인 시계열 데이터를 처리하는 데 뛰어난 성과를 보였다. 반면, GRU는 LSTM과 거의 비슷한 수준의 예측 정확도를 유지하면서도 계산 효율성 면에서 우위를 점하였다.

	ARIMA	LSTM NN	GRU NN
MSE	841.0065	710.0502	668.9304
MAE	19.1753	18.127758	17.2116

Table 2.4: MAE and MSE for Three Models(2016)

Zhang Mingheng, Zhen Yaobao, Hui Ganglong, Chen Gang(2013)은 SVM(Support Vector Machine)을 이용한 교통 예측용 인공 신경망을 제작하였다. 연구에서는 실제 교통량 데이터를 입력 벡터로 사용하여 SVM을 훈련하였고, 다양한 입력 벡터 조합[8]을 비교하여 가장 우수한 예측 성능을 보이는 모델을 도출하였다. 논문에서 제안된 다중 스텝 예측(multisteps prediction)은 기존의 단일 스텝 예측 모델에 비해 시간적 변화 추세를 예측하는 능력이 뛰어나다. 단일 스텝 예측이 현재의 교통 상태를 예측하는 데 중점을 둔 반면, 다중 스텝 예측은 향후 일정 기간 동안의 교통 상태 변화를 추정하는 데 유리하다. 이 모델은 시간적 자가상관과 과거 교통 패턴을 결합함으로써 교통 혼잡의 발생 가능성과 지속 시간을 보다 정확하게 예측할 수 있었다 제안된 SVM 모델은 기존의 통계적 모델보다 우수한 예측 성능을 보였으며, 특히 SVM-HPT 모델[9]이 가장 뛰어난 예측 성능을 발휘하였다 **??**. 이 모델은 다중 스텝 예측에서 시간적 변화와 공간적 상관성을 함께 고려하여 예측 정확도를 크게 향상시켰으며, 특히 혼잡 구간과 같은 복잡한 교통 패턴을 잘 포착하였다.

Yan-ni Yang, Hua-pu Lu(2010)은 SVM(Support Vector Machine)과 Wavelet

[8]시간적 자가상관 특징, 과거 패턴 데이터, 공간적 상관 데이터
[9]시간적, 공간적 상관 관계와 과거 패턴 데이터를 결합한 모델

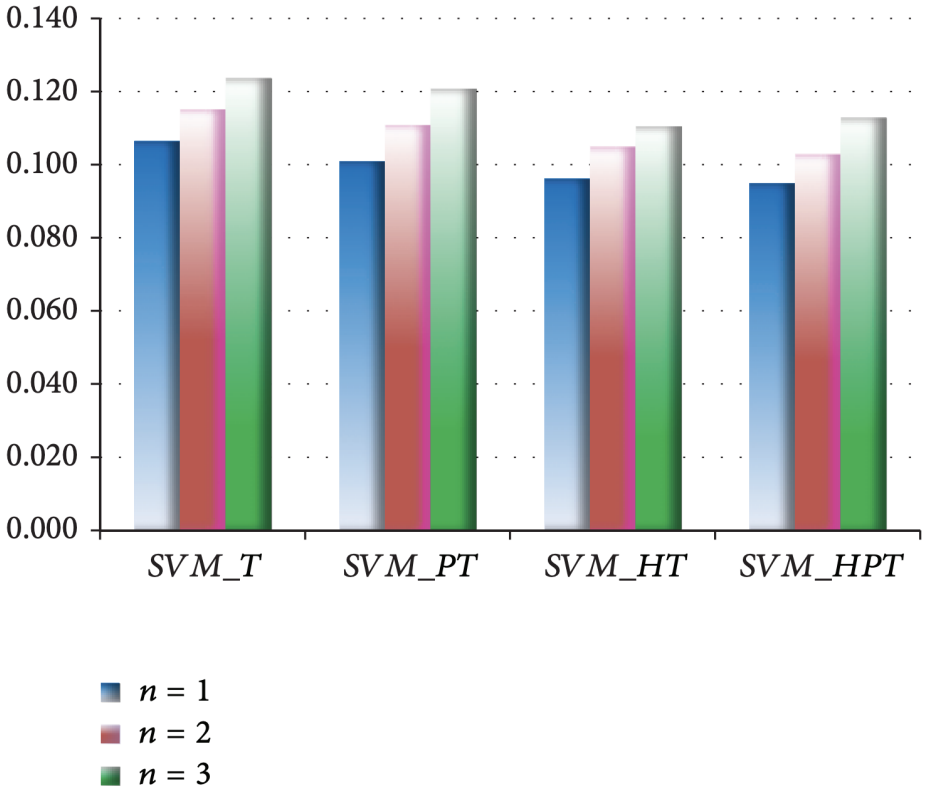

Figure 2.8: Prediction error results of the SVM model with different input vectors

Transform을 사용하여 지역 최적화, 과적합 및 고차원 문제 해결에 우수한 일반화 능력을 가진 신경망을 제작하였다. 웨이블릿 분석은 교통 흐름 데이터를 여러 주파수 성분으로 분해하여 본질적인 교통 패턴과 불확실한 간섭 신호를 분리하는 데 유용하다. 이를 통해 원래의 복잡한 시계열 데이터를 더 단순한 데이터 시퀀스로 변환할 수 있으며, 각 시퀀스는 SVM을 통해 개별적으로 예측되었다. 이러한 방식은 교통 흐름 데이터의 전반적인 변화 추세와 더불어, 예측 정확도를 저해하는 노이즈를 제거하는 데 효과적이었다. 멀티 해상도 분석의 원리에 따라, 교통 데이터는 기본 신호와 독립

적인 간섭 신호로 분해되며, SVM은 각각의 신호를 예측한 후 그 결과를 종합하여 최종 예측 값을 도출하였다. 연구에서는 가우시안 커널 함수를 사용하여 교통 데이터의 비선형적 관계를 고차원 특성 공간으로 매핑하고, 최적의 초평면을 구축하여 예측 성능을 극대화하였다. 또한, ϵ-인식 함수를 사용하여 실제 값과 예측 값 간의 오차를 측정하고, 이를 최소화하기 위한 최적화 과정을 거쳤다. 논문에서는 베이징 고속도로에서 수집된 실제 교통 데이터를 이용해 제안된 모델의 성능을 검증하였다. 제안된 웨이블릿-SVM 결합 모델은 기존의 단일 SVM 모델에 비해 예측 정확도가 유의미하게 향상되었으며, 특히 교통 흐름이 완만하게 변화할 때 높은 예측 성능을 보였다. 반면, 교통 흐름이 급격하게 변화하는 구간에서는 간섭 신호의 영향으로 인해 예측 정확도가 다소 떨어지는 경향을 보였다.

2.2.2 시계열 예측 기반 선행연구

MASCHAVAN DERVOORT, MARK DOUGHERTY, SUSAN WATSON(1996)은 교통 흐름 예측을 위해 코호넨 자기 조직화 맵(Kohonen Self-Organizing Map, SOM)과 ARIMA 시계열 모델을 결합한 하이브리드 예측 방법, 일명 KARIMA를 제안하였다. 교통 흐름의 단기 예측은 지능형 교통 시스템에서 중요한 역할을 하며, 다양한 알고리즘이 제안되어 왔으나, 이 연구에서는 데이터의 복잡성과 예측 성능 향상을 위해 계층형 모델 접근 방식을 채택하였다. 특히, 코호넨 맵을 활용하여 데이터를 효율적으로 분류하고, 각 클래스에 개별적으로 최적화된 ARIMA 모델을 적용함으로써 기존의 단일 모델보다 우수한 성능을 달성하였다. 코호넨 맵은 입력 데이터의 시각적 피드백을 제공하고, 클러스터링 과정에서 데이터를 구조적으로 분리하는 데 유리하다. 연구 결과, 프랑스 고속도로 데이터 세트를 대상으로 30분 및 1시간 예측 실험을 진행한 결과, KARIMA 모델은 적은 수의 클래스만으로도 높은 예측 정확도를 보였으며, ATHENA와 같은 복잡한 모델에 비해 실용적인 이점을 지닌 것으로 나타났다. 특히, 육각형 배열의 코호넨 맵을 사용한 경우 클러스터링 성능이

개선되어, 공간적 관계를 반영한 데이터 분류에서 더욱 효과적인 결과를 얻었다. 이러한 방식은 교통 흐름의 비선형적 특성을 잘 포착할 수 있으며, 계층형 구조를 통해 예측 성능을 극대화할 수 있다.

S. Vasantha Kumar, Lelitha Vanajakshi(2015)은 ARIMA모델에 계절성을 추가한 SARIMA모델을 사용하였다. 연구에서는 3일간의 제한된 데이터만을 사용하여 교통 흐름을 예측하는 SARIMA 기반의 모델을 제안하였다. 연구에 사용된 데이터는 인도의 첸나이(Chennai) 지역의 라지브 간디 도로에서 수집되었으며, 3차선 도로에서 3일간 수집된 데이터를 기반으로 모델을 구축하였다. 연구에서는 하루 24시간 동안 수집된 1분 단위의 교통량 데이터를 10분 간격으로 집계하여 총 144개의 데이터 포인트를 생성하였고, 이를 SARIMA 모델에 입력으로 사용하였다. SARIMA 모델은 교통 흐름에서 보이는 일일 주기성을 반영할 수 있도록 고안된 것으로, 특히 아침과 저녁의 피크 타임을 예측하는 데 있어 적합한 성능을 보였다. SARIMA 모델의 개발 과정은 모델 식별, 모델 추정, 진단 검사, 그리고 예측 결과의 검증 단계를 포함한다. 첫 번째 단계인 모델 식별에서는 시계열 데이터를 분석하여 계절성과 추세를 확인하고, 필요한 차분(differencing)을 적용해 데이터를 정상화하였다. 연구에서 사용된 데이터는 명확한 계절적 패턴을 보여, 계절적 차분을 통해 시간 간격이 144인 데이터로 변환되었다. 이후 자기상관 함수(ACF)와 부분 자기상관 함수(PACF)를 사용하여 적합한 SARIMA 모델의 차수를 결정하였다. 그 결과, 비계절적 차수는 AR(2), 계절적 차수는 MA(1)로 설정된 SARIMA(2,0,0)×(0,1,1)144 모델이 가장 적합한 것으로 확인되었다. 모델 추정 과정에서는 최대우도법(maximum likelihood method)을 사용하여 모델의 파라미터를 추정하였고, Akaike 정보 기준(AIC)을 통해 최적의 모델을 선택하였다. 검증 단계에서는 9월 23일의 교통 흐름을 예측하기 위해 9월 20일부터 22일까지의 데이터를 사용하여 예측을 수행하였으며, 예측값과 실제 관측값 간의 평균 절대 백분율 오차(MAPE)를 계산하였다. 그 결과, MAPE는 9.22%로, 교통 예측 연구에서 일반적으로 허용되는 10% 이하의 오차율을 기록하여

매우 높은 정확도를 보였다.

2.2.3 수리적 모델 기반 선행연구

Shiliang Sun(2004)은 불완전한 데이터를 활용한 단기 교통 흐름 예측을 목적으로, 고차 마르코프 체인 모델과 가우시안 혼합 모델(GMM)을 결합한 샘플링 마르코프 체인(Sampling Markov Chain) 방법을 제안하였다. 이 방법은 불완전한 데이터를 처리하는 기존 예측 모델들의 한계를 극복하고자, 상태 전이 확률을 GMM으로 근사한 후, 경쟁 기대 최대화(CEM) 알고리즘을 통해 최적의 파라미터를 추정하며, 불완전한 데이터에 대해서는 몬테카를로 통합(Monte Carlo Integration)을 이용해 샘플을 생성하여 최소 평균 제곱 오차(MMSE) 기준 하에 최적의 예측을 수행한다. 본 논문에서는 베이징 UTC/SCOOT 시스템에서 수집된 실제 데이터를 바탕으로 제안된 방법을 실험적으로 검증한 결과, 샘플링 마르코프 체인 방법이 불완전한 데이터 상황에서도 기존의 방법들에 비해 높은 예측 성능을 발휘함을 확인하였다.

Okutani, Iwao and Stephanedes, Yorgos J(1984)은 칼만 필터링(Kalman Filtering) 이론을 기반으로 단기 교통량을 예측하기 위한 두 가지 모델을 제안하였다. 기존의 교통량 예측 방법들은 일반적으로 히스토리컬 데이터에 의존하거나, 단일 링크에서의 데이터를 기반으로 예측을 수행하는 한계가 있었다. 논문에서 제안된 모델은 가장 최근의 예측 오차를 활용해 예측 파라미터를 개선하고, 단일 링크가 아닌 다수의 링크 데이터를 결합하여 예측 정확도를 향상시켰다. 칼만 필터는 교통 흐름 데이터의 단기적 비정상성을 처리할 수 있으며, 기존의 UTCS-2와 같은 2세대 예측 모델보다 최대 80% 더 나은 성능을 보였다. 특히 본 연구에서 제안된 모델은 나고야시의 도로망 데이터를 기반으로 검증되었으며, 평균 예측 오차가 9% 이하, 최대 예측 오차가 30% 이하로 측정되었다. 논문에서 제안된 두 가지 예측 모델 중 첫 번째 모델은 단일 링크의 교통량뿐만 아니라 인접 링크에서의 교통량 데이터를 함께 사용하여 15분 단위의 교통량을 예측한다. 이 모델은 현재 시점에서 미래의 교통량을 예측할

수 있도록 선형 예측 모델을 사용하며, 각 시점에서의 예측 오차를 고려하여 칼만 이득(Kalman Gain)을 통해 예측 값을 지속적으로 업데이트하였다. 두 번째 모델은 주간 간 교통량의 유사성을 고려하여, 한 주 전의 데이터를 기반으로 현재 주의 교통 흐름을 예측하였다. 이 모델은 특히 특정 요일의 교통 흐름 패턴이 전 주의 동일한 요일과 비슷하다는 가정 하에 정상성(stationarity)을 가정하고, 예측을 수행한다. 제안된 모델은 기존의 UTCS-2 알고리즘보다 더 나은 성능을 보였으며, 특히 교통량 데이터가 시간에 따라 빠르게 변동하는 경우에도 안정적인 성능을 유지하였다. 또한, 제안된 모델은 다중 스텝 예측(multi-step prediction)에서도 높은 정확도를 유지하였으며, 더 긴 시간 범위에서의 예측 성능 저하가 거의 발생하지 않았다.

2.2.4 국내 교통 예측 사례

서울특별시의 교통정보시스템인 TOPIS(서울교통정보센터, Traffic Operation Information Service)는 복잡한 도시 교통 환경을 효율적으로 관리하기 위해 구축된 종합 교통 관리 시스템으로, 첨단 정보기술과 빅데이터를 기반으로 실시간 교통 상황을 모니터링하고 미래 교통을 예측하는 핵심 역할을 수행하고 있다. 이 시스템은 도시 교통망의 체계적인 운영뿐만 아니라 돌발 상황에 대한 즉각적인 대응을 가능하게 함으로써 교통 혼잡을 완화하고 시민들에게 실시간으로 교통 정보를 제공하는 중요한 역할을 한다. TOPIS는 특히 도로, 버스, 지하철 등 다양한 교통수단의 정보를 통합하여 관리하며, 인공위성 기술을 활용한 버스 종합관리 시스템(Bus Management System, BMS)을 통해 서울시 전역의 버스 운행 데이터를 실시간으로 수집 및 분석한다. 이를 통해 버스 도착 시간 예측 서비스, 정류소별 도착 정보 제공 등 대중교통 이용자들의 편의를 증대시키고 있다. 이와 더불어, TOPIS는 기상 상황에 따라 도로 침수 및 교통 통제와 같은 예측 가능한 상황에 대해서도 사전 예보 서비스를 제공한다. 예를 들어, 집중호우 시 한강의 수위가 경계 수준에 도달하면 잠수교와 같은 상습 침수 구간의 교통 통제 예보를 사전에 알림으로써 시민들이 우회할 수 있도록

유도한다. 이러한 교통 통제 예보는 도로 전광표지(VMS), 내비게이션 앱 등을 통해 실시간으로 제공되며, 교통 혼잡을 사전에 방지하고 시민들의 안전을 확보하는 데 기여하고 있다. 특히, 서울시는 빅데이터와 교통 시뮬레이션을 결합한 첨단 기술을 통해 교통 상황을 예측하고, 대규모 집회나 행사로 인한 교통 통제 시 최적의 우회 경로를 제공하는 기능도 강화하였다. 이는 교통 시뮬레이션을 통해 도로망 및 차량 흐름을 모델링함으로써 교통 흐름을 예측하고 돌발 상황에 신속하게 대응할 수 있도록 하는 기술적 기반을 제공한다. TOPIS는 교통 관리뿐만 아니라 재난 관리까지 포괄하는 스마트 도시 관리 시스템(Smart Urban Management)으로 발전하고 있으며, 교통과 재난 정보를 통합하여 보다 안전하고 효율적인 도시 운영을 지원하고 있다. 이는 서울시가 교통관리뿐만 아니라 재난 대응에도 적극적으로 활용하는 중요한 도시 관리 인프라로 자리 잡고 있다.

2.2.5 국외 교통 예측 사례

구글 딥마인드(DeepMind)는 교통 예측 분야에서 혁신적인 인공지능(AI) 기술을 통해 구글 지도(Google Maps)의 실시간 예상 도착 시간(ETA) 예측 성능을 크게 향상시켰다. 딥마인드는 그래프 신경망(Graph Neural Networks, GNN)을 활용하여 도로망을 그래프로 모델링하고, 이를 통해 교통 흐름을 더욱 정확하게 예측하고 있다. 이 방식은 특히 복잡한 도로 네트워크와 시간적 변동성을 효과적으로 처리할 수 있어, 기존 예측 시스템 대비 정확도를 최대 50%까지 향상시키는 결과를 도출하였다. 주요 개선 지역으로는 베를린, 자카르타, 상파울루, 시드니, 도쿄, 워싱턴 D.C. 등이 있다. 구글 지도는 이전에도 실시간 교통 데이터를 바탕으로 예측을 수행했으나, 딥마인드의 GNN 기술을 도입함으로써 미래의 교통 상황을 예측하는 능력이 대폭 향상되었다. GNN은 각 도로 구간을 '슈퍼세그먼트(Supersegment)'라는 단위로 나누고, 이들 간의 관계를 모델링하여 다양한 시간대의 교통 흐름을 예측한다. 이러한 시스템은 도로의 품질, 제한 속도, 사고 및 폐쇄 등의 다양한 변수들을 통합적으로

고려할 수 있어, 교통 혼잡 발생 가능성 등을 사전에 경고할 수 있다. 또한, 딥마인드의 교통 예측 모델은 인공지능의 학습 안정성을 높이기 위해 메타그래디언트 (MetaGradients) 등의 기술을 도입하여, 다양한 환경에서 정확한 예측을 가능하게 하였다. 이러한 기술은 전 세계적으로 적용할 수 있는 확장성을 보유하고 있어, 교통 예측뿐만 아니라 다양한 실시간 데이터 기반 서비스에 유용하게 활용될 수 있다.

2.2.6 기존 연구와의 차별성

1) 수리적 교통류 이론들과의 결합

기존의 선행연구들을 수리적, 비모수적, 그래프적, 시계열 예측의 방법으로 나뉘어 각자의 분야에서 모델을 만들고, 교통 흐름을 예측하였다. 또한 모델과 모델, 모델과 이론들의 결합을 시도할 때도 각 분야의 범위 내에서 시행되고 있는 실정이다. 그러나 본 연구에서는 충격파 이론, 뉴웰의 관성모델 등의 수리적 접근법과 비모수적 접근의 일부인 딥러닝 기법을 결합시켰다. 이에 더불어 본 연구에서는 단순한 수식의 정립에서 끝나는 것이 아닌, 실제 모델을 구현하고 그 성능을 검정하였다.

2) 대한민국의 교통 데이터 사용

기존의 국외 선행 연구에서는 일반적으로 도시 단위의 교통 데이터를 사용하여, 특정 도시의 지점 정보를 집중적으로 분석하는 형태를 보였다. 또, 이런 데이터에 그 지역과 국가의 고유 정보가 부족하여 예측 정확도도 떨어졌다. 반면 본 연구에서는 대한민국의 표준노드링크, 소통 및 돌발 정보, 기상 정보같은 대한민국 고유의 정보를 사용할 뿐만 아니라 그 데이터의 범위도 대한민국 전역으로 매우 넓다.

3) 다양한 기법의 적용

본 연구에서는 데이터 전처리 단계에서 DASK, geopandas, Missingo등의 최신 데이터 처리 패키지들이 다수 사용되었다. 또 모델의 개발과 학습 단계에서 연속 학습, 전이 학습 등의 고급 학습 기법들이 적용되었다. 마지막으로, 본 연구는 예측된 데이터를 분석하는 과정에서 GIS 프로그램 등의 지리적 시각화 기법을 사용해서 더욱 직관적인 고찰이 가능하다.

위 차별점들을 통해 본 연구는 기존 연구보다 더욱 발전된 교통 예측을 가능하게 하였다.

3장. 모델 구축 방향

세가지 교통류 이론과의 결합 모델에 공통적으로 적용될 구축 방향을 살펴보고자 한다. 구축 방향에는 세가지 모델에 공통으로 입력할 데이터의 요구 유의 변수, 템플릿 등을 정의하고자 한다. 또 세가지 모델의 기반 모델에 대한 고찰과 특성을 살펴보고자 한다.

3.1 기반 모델 선정

본 연구는 기반 모델로서 세가지 모델에 공통으로 GRU, 그리고 각 모델에 LWR의 충격파 이론[10], 뉴웰의 관성 모델, 그리고 다상 교통 이론을 사용하였다. 본 절에서는 각 기반 모델의 특성을 분석하고, 이를 통해 선정 이유를 규명하고자 한다.

3.1.1 GRU

본 연구에서는 세가지 모델에 공통적인 기반 모델로 GRU모델을 사용하였다. 다음은 GRU모델의 원식이다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

[10]충격파 이론의 대부분은 LWR의 이론을 응용하였지만, 그외 후속 연구에서 정립된 다양한 이론들을 시도하였다.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$$

$$\tilde{h}t = \tanh(W_h \cdot [r_t \odot ht - 1, x_t] + b_h)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

1) 예측 정확도

GRU모델은 기존의 몇가지 선행연구를 통해 기존 딥러닝 모델인 SVM, rnn에 비해 월등한 예측 정확도를 지녔으며, 또 기울기 폭주와 소실같은 학습 시 문제또한 존재하지 않는다는 것이 입증되었다. GRU모델은 은닉 상태를 사용하여 비선형성과 돌발 상황에 대한 학습 성능을 증가시켰고, 또 데이터의 과적합 현상을 줄였다.

2) 연산 속도

연산 속도는 GRU의 최대 강점이라고 분석할 수 있다. 위에서 얘기한 예측 정확도는 또 다른 순환신경망인 LSTM과 GRU가 비슷한 수치를 보인다. 그러나 LSTM은 셀, 입력 게이트, 출력 게이트, 망각 게이트 등의 여러 개의 단계를 거쳐 예측과 학습 시의 연산 속도가 상대적으로 느린 편이다. 그러나 GRU는 은닉 상태만을 이용하여 매우 빠른 연산 속도를 보인다. LSTM와 GRU의 연산을 Big O notaion[11]으로 나타내면, $O((d+h) \cdot h)$[12] 으로 동일하다. 하지만 이것은 게이트의 수를 고려하지 않고 계산한 값이고, 만약 더욱 자세한 시간 복잡도를 위해 상수를 표기하게 되면 다음과 같다. LSTM의 상수를 포함한 Big O notation은 $O(4(d+h) \cdot h)$고, GRU의 상수를 포함한 Big O notation은 $O(3(d+h) \cdot h)$ 이다.

[11]Big O notation은 알고리즘의 성능을 입력 크기에 따라 수학적으로 분석하기 위한 도구로서, 주어진 알고리즘의 최악의 실행 시간을 표현하는 데 주로 사용된다. 이는 알고리즘의 효율성을 평가할 때 가장 중요한 척도 중 하나로, 입력 데이터의 크기가 무한히 커질 때 알고리즘의 실행 시간이 어떻게 변화하는지를 나타낸다. Bit O notation은 함수의 상수 계수나 낮은 차수 항을 무시하고, 가장 빠르게 증가하는 항만을 남겨 시간 복잡도나 공간 복잡도를 간결하게 표현함으로써 알고리즘의 상대적 성능 비교를 가능하게 한다.

[12]입력 크기: d, 은닉 상태 크기: h

3) 간결한 구조

위 연산 속도 항목에서도 설명하였지만, GRU는 리셋 게이트와 업데이트 게이트의 두개의 게이트만 사용한다. 따라서 매우 적은 수의 파라미터 수만을 사용한다. 이는 결합 모델의 설계 시 고려할 게이트와 파라미터의 수가 적어짐을 의미하고, 이는 모델 설계 시 수정이 자유롭게 한다. 또 특정 이론을 적용할 때 은닉 상태와 입력에만 수식을 적용시키면 되기에 각 이론의 특징을 선명하게 반영할 수 있다.

4) 성능 검증

위 세개의 이유들에 의해 본 연구에서는 GRU를 세가지 결합 모델의 공통 기반 모델로 선정하였다. 다음은 다른 순환 신경망인 RNN, LSTM과 GRU의 성능을 비교한 지표이다. 사용된 데이터는 NIFTY-50 Stock Market Data (2000 - 2021)의 데이터셋에서 Bharat Petroleum Corporation Ltd(BPCL)의 주가 데이터를 사용하였다 [13]. 예측 목표 데이터는 종가를 사용하였다. 최소-최대 정규화(Min-Max regularization)을 사용하여 데이터를 정규화하였다.

$$X_{\text{scaled}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}(\max - \min) + \min$$

훈련 데이터와 테스트 데이터를 80%와 20%의 비율로 분할하였다. 두개의 노드를 사용하였고, 각 노드당 50개의 유닛을 구축하였다. 학습률 0.001의 Adam Optimizer를 사용하였고, MSE를 손실 함수로 사용하였다. 마지막으로 에포크는 100, hidden size는 32이다. 평가 지표는 MSE, RMSE, MAE, R^2를 사용하였다 **??**. 각 지표는 소수점 다섯째 자리에서 반올림하였다.

[13] `https://www.kaggle.com/datasets/rohanrao/nifty50-stock-market-data`

	GRU	LSTM	RNN
Train MSE	243.051	244.8846	345.8985
Train RMSE	15.5901	15.6488	18.5983
Train MAE	8.3855	8.5409	12.3286
Train R^2	0.9939	0.9939	0.9913
Test MSE	158.9586	163.5484	230.6292
Test RMSE	12.6079	12.7886	15.1865
Test MAE	7.8448	8.0651	11.1789
Test R^2	0.9857	0.9853	0.9793

Table 3.5: 세가지 순환신경망 모델 비교

R^2지표는 비슷한 정도이지만, 그외 모든 오차 지표에서 GRU가 가장 낮은 오차률을 보였다. 각 모델의 시간별 Loss 추이는 다음과 같다 **??**, **??**, **??**.

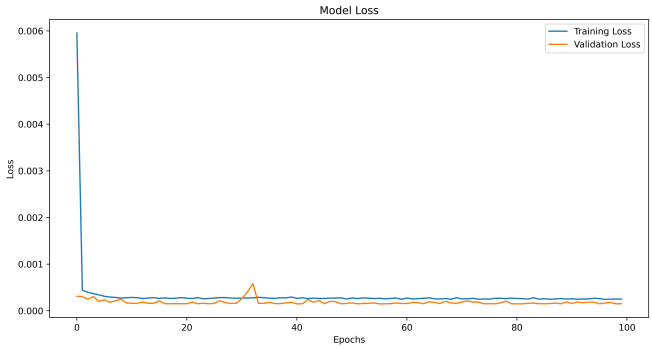

Figure 3.9: GRU Model Loss

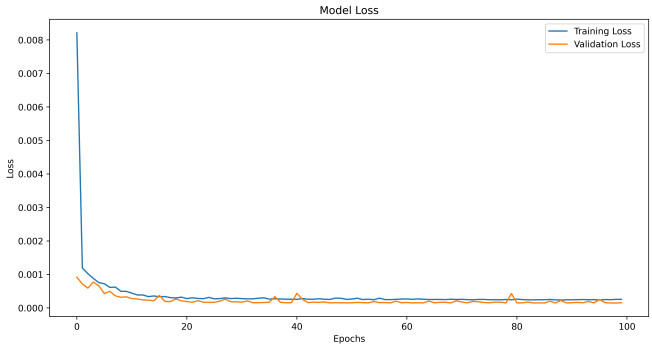

Figure 3.10: LSTM Model Loss

3.1.2 충격파 이론(LWR)

충격파 이론(Shockwave Theory)은 교통류 분석에서 교통 혼잡의 전이와 급격한 변화, 즉 교통 흐름의 불연속적인 변화를 설명하는 중요한 이론적 틀이다. 이 이론은 물리학의 유체 역학에서 유래했으며, 교통류에서 차량 밀도, 속도, 흐름 간의 상호작

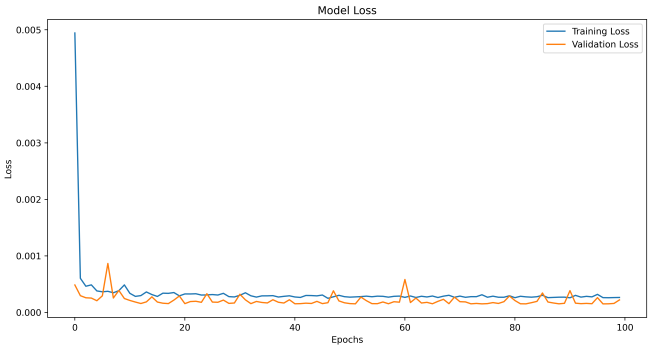

Figure 3.11: RNN Model Loss

용을 분석하는 데 적용된다. 교통류에서의 충격파는 교통 밀도의 급격한 변화로 인해 발생하며, 차량 간 간격이나 속도에서의 변화가 충격파를 형성한다. 이는 보통 선두 차량이 감속하거나 정지할 때 뒤따르는 차량들이 연쇄적으로 속도를 줄이면서 발생한다. 이러한 충격파는 도로 상에서 이동하는 밀도 파동처럼 작용하며, 교통 정체가 뒤로 확산하는 원인을 설명하는 데 사용된다.

1) 운동파 이론

M. J. Lighthill, G. B. Whitham(1955)은 교통 흐름을 유체역학적 관점에서 모델링하는 운동파 이론(Kinematic Wave Theory) 혹은 LWR모델을 제시하며, 교통 흐름을 설명하기 위해 연속 방정식과 충격파 이론(shockwave theory)을 도입한 연구다. M.J. Lighthill과 G.B. Whitham은 도로 위에서 차량의 움직임을 유체의 흐름과 유사하게 취급하였고, 이를 통해 도로 내에서 차량의 밀도와 속도 변화를 설명하였다. 이들의 모델은 교통 혼잡과 같은 비선형적 현상을 물리적으로 설명할 수 있는 이론적 기반을 마련하였으며, 교통 흐름의 시간적, 공간적 변화를 설명하기 위해 수학적 방정식을 활용하였다.

논문은 교통 흐름을 분석하기 위해 연속 방정식(continuity equation)을 도입한다. 이는 일정한 시간 동안 특정 도로 구간에 차량이 추가되거나 빠져나가는 차량 수와 교통 밀도, 차량 속도 간의 관계를 설명하는 방정식이다. 이 방정식은 다음과 같은 형태로 나타낼 수 있다.

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} = 0$$

여기서, ρ는 교통 밀도(단위 거리당 차량 수), u는 차량의 속도, x는 위치, t는 시간이다. 이 방정식은 교통 흐름이 시간과 공간에 따라 어떻게 변화하는지를 기술하며, 유체역학에서의 연속성 방정식과 유사하다.

교통 흐름의 변화를 설명하기 위해 Lighthill과 Whitham은 파동 속도(wave speed) 혹은 충격파 속도(shockwave speed) 개념을 도입하였다. 교통 밀도가 변할 때 그 변화가 도로를 통해 어떻게 전파되는지를 설명하기 위해, 밀도의 변화가 도로 상에서 이동하는 속도인 파동 속도를 정의하였다. 파동 속도 w는 다음과 같이 정의된다.

$$w = \frac{\Delta Q}{\Delta \rho}$$

여기서 Q는 교통량(단위 시간당 차량 수)이다. 이는 밀도가 변할 때 그 영향이 도로 상에서 전파되는 속도로, 교통 밀도가 증가하거나 감소함에 따라 파동이 이동하는 방식을 설명한다. 이 파동은 혼잡이 발생할 때 충격파(shockwave)로 나타나며, 이는 교통량이 갑작스럽게 변화하는 상황에서 혼잡 구간이 어떻게 형성되고 확산되는지를 설명한다. 충격파의 속도는 교통 밀도의 변화율에 따라 결정되며, 이 속도가 음수일 경우 교통 정체가 뒤로 전파되는 특성을 가진다. 논문에서 중요한 또 하나의 개념은 기본 다이어그램(fundamental diagram)으로, 이는 교통량과 밀도 간의 관계를 설명하는 그래프이다. Lighthill과 Whitham은 교통 흐름이 일정한 속도 범위 내에서는 교통량이 증가하지만, 특정 밀도 이상에서는 교통량이 감소하는 현상을 발견하였다.

이는 도로 용량이 초과되면 교통량이 더 이상 증가하지 않고 오히려 혼잡이 발생하며, 속도가 줄어들고 차량의 흐름이 느려지는 비효율적인 상태가 된다.

이 이론은 교통류의 속도 변화가 변화하는 방식을 다양한 관점에서 설명할 수 있기 때문에 결합 모델로 사용하였다.

3.1.3 뉴웰의 관성 모델

뉴웰(G. F. Newell)의 차량 추종 모델은 교통 흐름의 비선형성을 설명하기 위해 수학적 접근 방식을 취하며, 운전자가 앞차와의 거리(헤드웨이)와 속도 차이에 어떻게 반응하는지를 모델링한다. 이 모델의 핵심은 반응 지연 시간(Reaction Delay), 속도 변화, 차량 간 간격을 기반으로 한 비선형적 관계를 통해 차량의 동역학을 기술하는 것이다.

1) 비선형 속도 반응 함수

뉴웰은 운전자가 앞차와의 거리 및 속도 차이에 따라 반응하는 방식을 비선형 속도 함수로 표현하였다. 이 비선형 함수는 앞차와의 상대 위치와 상대 속도에 따라 결정되며, 이를 통해 차량이 어떻게 가속 또는 감속하는지를 설명한다. 수학적으로, 뉴웰의 모델은 다음과 같은 형태로 정의된다:

$$v_j(t) = G\left(z_{j-1}(t-\Delta) - z_j(t-\Delta)\right)$$

- $v_j(t)$는 시간 t에서 j번째 차량의 속도
- $z_{j-1}(t-\Delta)$는 앞차의 위치(시간 지연 Δ를 고려한)
- $z_j(t-\Delta)$는 j번째 차량의 위치(시간 지연을 고려한)
- $G(\cdot)$는 비선형 함수로, 속도 변화에 대한 운전자의 반응

이 함수는 차량 간의 간격과 속도 차이에 따라 차량의 속도가 어떻게 조정되는지를 결정하며, 반응 시간(Reaction Time, Δ)이 모델의 중요한 변수로 작용한다.

2) 시간 지연과 가속도

뉴웰은 차량 추종에서의 가속도(acceleration)와 감속도(deceleration)가 비선형적으로 이루어짐을 강조했다. 즉, 차량은 앞차와의 거리 및 속도 차이에 따라 일정한 비율로 속도를 조절하는 것이 아니라, 변화하는 상황에 따라 다양한 비율로 가속 또는 감속한다. 이는 교통 혼잡 상황에서 차량 간의 상호작용이 단순히 거리에 기반한 선형 모델로는 설명되지 않음을 보여준다.

가속도 방정식은 차량의 속도가 앞차와의 간격 변화에 따라 비선형적으로 변한다는 사실을 반영하며, 이를 통해 차량이 교통 흐름에서 어떻게 반응하는지를 설명한다.

$$a_j(t) = k \cdot \left(\frac{1}{z_{j-1}(t) - z_j(t)} - \frac{1}{\text{desired gap}} \right)$$

여기서:

- $a_j(t)$는 j번째 차량의 가속도
- k는 반응 계수(Constant)
- $z_{j-1}(t) - z_j(t)$는 앞차와의 거리(헤드웨이)
- desired gap은 차량 간의 이상적인 간격

위 내용처럼 뉴웰의 관성 모델은 차량 간의 미시적인 충격파의 전달이 시간 지연에 따라 진행된다는 이론을 설명한다. 이 이론들이 본 연구에서 개발하고자 하는 결합 모델에서 미시적 예측에 효율적일 것이라 판단하였다.

3.1.4 다상 교통 이론

다상 교통 이론은 B. S. Kerner가 두편의 논문에서 발표한 교통 이론으로, 3가지의 상태로 구성되어있는 다상 이론이다. 이 이론은 각 상태의 전이 관계를 설명하며, 이는 교통 예측에 긍정적인 영향을 미칠 수 있다.

1) 교통 흐름의 상전이와 자가 조직화

Kerner는 교통 흐름에서 두 가지 주요한 상전이 현상을 발견했다. 첫 번째는 자유 흐름(free flow)에서 동기화 흐름(synchronized flow)으로의 상전이이고, 두 번째는 동기화 흐름에서 정체 흐름(jam)으로의 상전이다. 이 상전이는 특정 차량 밀도 범위에서 발생하며, 이러한 밀도 범위에서는 더 이상 균일한 상태를 유지할 수 없기 때문에 자연스럽게 차량의 속도와 밀도가 변동하게 된다. Kerner는 특히 교통 흐름에서의 비선형적인 특성, 즉 작은 변동이 시간이 지남에 따라 큰 혼잡으로 이어지는 과정을 수학적으로 설명하였다. 교통 흐름에서의 상전이 과정은 교통 밀도가 낮을 때는 자유 흐름이 유지되지만, 밀도가 증가하면 차량 간의 상호작용이 강화되어 동기화된 흐름으로 전환된다. 이 동기화 흐름은 불안정한 상태로, 추가적인 밀도 증가에 따라 좁은 정체(narrow jams)가 발생하고, 시간이 지나면서 이러한 좁은 정체가 점차 넓은 정체(wide jams)로 확장된다. Kerner는 이러한 정체가 도로 상에 일정한 간격을 두고 반복적으로 형성되는 현상을 자가 조직화(self-organization)로 설명한다.

2) 핀치 효과(Pinch Effect)

핀치 효과는 동기화 흐름 내에서 좁은 정체가 자발적으로 형성되고 성장하는 현상이다. 좁은 정체는 교통 흐름의 압축(compression)으로 인해 발생하며, 이 과정에서 차량의 속도는 줄어들고 밀도는 증가한다. Kerner는 좁은 정체가 형성된 후, 이러한 정체가 도로 상에서 어떻게 성장하고 합쳐져서 넓은 정체를 형성하는지 설명하였다.

핀치 영역(pinch region)에서 좁은 정체는 상호작용을 통해 넓은 정체로 변환되며, 이 과정에서 도로 상에 일정한 간격을 두고 Stop-and-Go 패턴이 나타난다. Kerner는 이러한 패턴이 형성되는 과정을 실험적으로 분석하였으며, 좁은 정체의 평균 간격은 차량 속도가 증가할수록 커진다는 사실을 발견했다. 이 결과는 실험 데이터를 기반으로 한 수치 모델링을 통해 확인되었으며, 정체 간의 거리는 차량 속도에 따라 비례적으로 증가한다는 결론을 도출하였다. Kerner는 이러한 현상을 설명하기 위해 밀도-흐름(density-flow) 평면에서 교통 상태의 변화를 시각적으로 설명하였으며, 이를 통해 정체의 형성과 소멸 과정을 정량적으로 분석하였다.

다상 교통 이론은 모두 같은 가중치를 통해 비선형성을 예측하는 기존 딥러닝 기법에 더욱 다양한 해석을 가능하게 하여 예측 정확도를 올릴 수 있다.

3.2 데이터 템플릿

본 연구에서 개발하고자 하는 세가지의 결합 모델은 모두 다변량 데이터를 활용한 교통 예측 모델이다. 따라서 세가지 모델 모두 공통된 데이터를 필요로 하며, 이는 정확한 성능 검정에도 필수적이다. 성능 비교에 사용될 모든 기존의 교통 예측 모델에도 같은 템플릿을 사용한다. 데이터 템플릿에는 요구 유의 변수, 데이터의 Shape 등이 포함된다. 데이터 템플릿은 본 연구에서 개발하고자 하는 세가지 결합 모델의 목적과 특성을 반영하여, 예측 성능을 가장 폭발적으로 이끌어낼 수 있게 구성하였다. 데이터 템플릿에서 요구하는 데이터의 기준은 본 연구에서 도출할 결과를 위해 최소한으로 필요하다.

3.2.1 출력 변수

본 연구에서 개발하고자 하는 교통 예측 모델은 교통량, 시간평균속도(Time Mean Speed, TMS)를 예측한다. 교통량은 단위시간당 조사지점을 지나간 차량의 수를 뜻하며, 본 연구에서 뜻하는 교통류는 시간별 교통량 변화를 뜻한다. 시간평균속도는 속도(u)로 표현되며, 지점속도라고도 표현된다.

$$\frac{1}{N}\sum_{i=1}^{N} u_i$$

3.2.2 요구 유의 변수 설정

교통류 예측을 위해서는 크게 교통류 직접 변수, 도로 환경 변수, 기상 환경 변수, 교통 수요 공공 변수, 차량 현황 변수 등이 필요하다. 다음은 각 유의 변수의 선정 사유와 변수의 자세한 설정이다.

1) 교통류 직접 변수

교통류 직접 변수(Direct Variable)는 교통류 예측 및 분석에서 예측 결과에 직접적으로 영향을 미치는 요인으로, 예측하고자 하는 목표 변수에 즉각적이고 명확한 영향을 주는 변수를 의미한다. 본 연구에서는 예측하고자 하는 출력은 q_t(시간평균 교통량, 유량), u_t(시간평균속도)이기 때문에 직접 변수는 q_{t-1}과 u_{t-1}가 포함된다. 본 연구에서 사용한 교통량(시간 당 통과한 차량의 수)의 단위는 vph(Veh/h, 대/시)를 사용하였다 **??**. 시간평균 속도의 단위는 일반적인 km/h를 사용한다.

2) 도로 환경 변수

도로 상황은 교통류에 큰 영향을 미치는 중요한 자료이다. 도로 환경 변수는 도로의 구성요소인 도로 길이, 도로 폭, 차로 개수, 도로 위치가 포함되어야 한다 **??**.

Variable	Unit
교통량(Q)	vph
속도(u)	km/h

Table 3.6: 교통류 직접 변수

가. 도로길이

도로 길이는 길어질수록 교통 혼잡이 줄어드는 경향을 띄는 간접 변수이다. 도로 길이는 데이터의 종류에 따라 단위의 통일이 어렵기 때문에 도로의 시작점과 끝 지점을 위, 경도 데이터로 받는다. 하지만 일반적인 위, 경도 데이터 사이의 거리를 구하기 위해 유클리디안 거리(Euclidean distance)를 산출하게 된다면 몇가지 문제가 발생한다.

먼저, 지구는 구형이므로 위도와 경도의 차이를 단순히 유클리디안 거리로 계산하면 지구의 곡률을 반영하지 못하며, 특히 경도는 위도에 따라 그 거리가 다르기 때문에(적도에서는 경도 1도가 더 길고, 극지방에서는 더 짧아짐) 정확한 실제 거리를 구할 수 없다 [14]. 또 위도와 경도의 단위는 도(°)이기에 유클리디안 거리의 단위도 도(°)가 되버리게 되어 거리 데이터로서 적합하지 않다.

이에 따라 본 연구에서는 거리 데이터를 계산하기 위해 하버사인 공식(Haversine Formula)을 사용하였다. 하번사인 공식은 지구를 완벽한 구형으로 가정하여, 지구 곡면상 두 지점 간의 대권(Great Circle) 거리를 구하는 공식이다. 하버사인 함수는 구면 좌표에서 각도 차이가 작을 때 발생할 수 있는 부정확성을 해결하여 작은 각도 차이에서도 높은 정확도를 유지하며, 구면 좌표계에서의 거리를 계산하는 표준 도구로 사용되고 있다. 지구 상 두 좌표가 주어졌을 때 거리를 구하는 하버사인 공식은

[14] 지구의 곡률은 1km당 약 7.85미터이다. 이는 지구의 평균 반지름인 6,371km을 기준으로 계산되었다.

다음과 같다.

$$d = 2r \cdot \arcsin\left(\sqrt{\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

- d: 두 지점 사이의 구면 거리(보통 킬로미터로 계산)
- r: 지구의 반지름 (평균 약 6371km)
- ϕ_1, ϕ_2는 첫 번째와 두 번째 지점의 위도(라디안 단위)
- λ_1, λ_2는 첫 번째와 두 번째 지점의 경도(라디안 단위)

예를 들어 지점1은 위도 52.2296756, 경도 21.0122287, 지점2은 위도 41.8919300, 경도 12.5113300로 설정한다. 하버사인 공식에 적용한다면 다음과 같다.

예) 위도와 경도를 라디안으로 변환

- $\phi_1 = 52.2296756 \times \frac{\pi}{180} \approx 0.911$, $\lambda_1 = 21.0122287 \times \frac{\pi}{180} \approx 0.367$
- $\phi_2 = 41.8919300 \times \frac{\pi}{180} \approx 0.731$, $\lambda_2 = 12.5113300 \times \frac{\pi}{180} \approx 0.218$

하버사인 공식 적용

$$d = 2 \times 6371 \times \arcsin\left(\sqrt{\sin^2\left(\frac{0.731 - 0.911}{2}\right) + \cos(0.911)\cos(0.731)\sin^2\left(\frac{0.367 - 0.218}{2}\right)}\right)$$

$$d \approx 2 \times 6371 \times \arcsin(0.1029) \approx 2 \times 6371 \times 0.1031$$

$$d \approx 1314.59\,\mathrm{km}$$

나. 도로 폭

도로 폭은 넓을 수록 교통 혼잡이 줄어드는 간접변수로, 대부분의 도로 데이터에 포함되어있다. 단위는 m를 사용한다.

다. 차로 개수

차로 개수는 많을수록 교통 혼잡이 줄어드는 간접변수로, 이 또한 도로 데이터에 포함되어있는 경우가 일반적이다. 단위는 (개)를 사용한다.

라. 도로 위치

도로 위치는 기상 데이터와 공공 데이터와 병합을 위해 필수적인 변수이다. 변수의 좌표계는 epsg:4326을 사용하여 일반적인 위, 경도 단위로 변환한다. 도로의 위치는 시작 지점와 끝 지점의 geometry가 제공될텐데, 이때 계산의 용이성을 위해 시작 지점의 위치 데이터를 사용한다.

Variable	Unit
도로 길이	m
도로 폭	m
차로 개수	개
도로 위치	도(°)

Table 3.7: 도로 환경 변수

3) 기상 환경 변수

기상 환경 변수는 기압, 온도, 풍향, 풍속, 강수량 등 다양한 요인이 있지만 교통류에 크게 영향을 주는 변수는 온도와 강수량이라고 할 수 있다. 따라서 본 연구에서

개발할 세가지 결합 모델에서 요구되는 필수 기상 변수는 온도와 강수량이다. 온도는 단위로 섭씨(°C)를 사용하였고, 강수량은 mm단위를 사용한다. 기상은 관측 지점에 따라 천차만별의 양상을 보인다. 기상 관측 지점 데이터는 관측 기관의 지점 구분 변수를 통해 사용하며, 지점의 위치는 위, 경도 데이터를 활용한다. 관측량을 사용할 지점 선정은 도로위치 변수와의 유클리디안 거리를 사용하여 최단 거리에 위치한 지점으로 선정한다[15].

Variable	Unit
온도	°C
강수량	mm
지점 위치	도(°)

Table 3.8: 기상 환경 변수

4) 교통 수요 공공 변수

공공 변수는 인구 데이터 하나만을 사용하였다. 인구 분포의 단위는 시군구 데이터를 사용한다. 나이 분포는 실질 운전자의 대부분이 위치하고 있는 50세 이하의 5세 단위를 사용한다. 인구 분포는 각 나이 별로 만 나이 단위를 사용하며, 인구수는 (명)를 사용한다. 병합 시 사용될 시군구는 각 시군구의 기준 위, 경도와의 유클리디안 거리를 통해 최단 거리 시군구를 사용한다[16]. 인구 데이터의 합계는 포함하지 않는다.

[15]기상 관측 지점은 대한민국 내에서 존재하고, 간격 간의 평균 거리가 유클리디안 거리의 곡률에 의한 오차 범위 안이다. 따라서 유클리디안 거리를 사용하였다.

[16]본 연구의 Case Study 각 시군구의 기준 위, 경도는 시, 군, 구청의 위치를 사용한다.

Variable	Unit
인구 수	(명)
나이 분포	명/세

Table 3.9: 교통 수요 공공 변수

5) 차량 현황 변수

차량 현황 변수는 교통 수요 공공 변수와 수요의 관점에서 교집합을 가지는 변수의 집합이다. 차량 현황 변수는 차량의 종류별 등록 현황과 전체 등록 현황, 운전면허 소지자 통계를 사용한다. 각 변수들은 모두 교통류에 직접적인 영향 보단 거시적 관점에서 윤곽이 드러나는 간접 변수이다. 차량의 종류별 등록 현황은 각 자동차 종류에 대해 (대)를 사용한다. 전체 등록 현황 또한 (대)를 단위로 사용하고, 운전면허 소지자 통계는 (명)을 사용한다.

Variable	Unit
차량 종류별 현황	(대)
차량 전체 분포	(대)
운전면허 소지자 분포	(명)

Table 3.10: 차량 현황 변수

3.2.3 데이터의 Shape

위에서 설정한 요구 유의 변수를 바탕으로 다음과 같은 데이터의 Shape과 Dtype을 가질 수 있다 **??**. 실제 데이터에는 시계열 데이터를 위한 Date 변수가 추가된다. 또 각 데이터의 dtype은 변환 시 자료 손실이 없는 반경에서 허용된다.

항목	설명
교통량	단위 시간 당 관측 지점을 통과한 차량의 수(단위: vph)
속도	시간 평균 속도(단위: km/h)
도로 길이	도로 시작과 끝 지점의 대권 거리(단위: m)
도로 폭	도로의 가로 폭(단위: m)
차로 개수	차로의 개수(단위: 개)
도로 위치	도로 시작 지점의 위, 경도(단위: 도)
온도	최단 거리 지점의 관측 온도(단위: 섭씨)
강수량	최단 거리 지점의 관측 강수량(단위: mm)
지점위치	최단 거리 관측 지점의 위, 경도(단위: 도)
인구 수	관측 시군구의 인구 수(단위: 명)
나이 분포	관측 시군구 인구의 나이 분포(단위: 명/세)
차량 종류별 현황	관측 날짜의 차량 종류별 등록 현황(단위: 대)
차량 전체 분포	관측 날짜의 차량 전체 분포(단위: 대)
운전면허 소지자 현황	관측 날짜의 운전면허 소지자 현황(단위: 명)

Table 3.11: 전체 요구 유의 변수 단위

Date	속도	(요구 유의 변수)	교통량
2023-08-02 00:00:00	45.5	...	4
2023-08-02 00:05:00	66.3	...	6
2023-08-02 00:10:00	55.4	...	7
2023-08-02 00:15:00	74.6	...	18
2023-08-02 00:20:00	22.4	...	20
...	...	...	...
dtype: timestamp/string/object	dtype: float64	dtype: NaN	dtype: int64

Table 3.12: 전체 요구 유의 변수 형상

4장. 충격파 이론 결합 모델 개발

4.1 충격파 이론의 특성 추출

충격파 이론에서는 다양한 관점에서의 해석이 존재하지만, 본 연구에서는 딥러닝 기법에 적용하기 적합한 일부 개념 들만을 추출하여 적용시켰다. 각 개념들은 모식적, 수식적으로 딥러닝 학습에 맞게 적절히 변환되어야 한다.

충격파 이론 결합 모델 개발의 전체 도식은 다음과 같다 **??**.

4.1.1 충격파 이론의 개념 사용

충격파 이론의 개념은 특정 지점에서 발생한 속도와 교통량의 변화가 주변으로 퍼져나간다는 현상이다. 즉, 입력 상태에 특정 변화가 일어날 시 이것이 일련의 과정을 거쳐 미래 상태에 영향을 주는 것으로 해석할 수 있다. 이는 딥러닝 모델에서 가중치가 입력 값에 영향을 주는 과정과 비슷한 개념으로 볼 수 있다. 이러한 판단에 의거하여 충격파 이론 결합 모델에는 GRU 모델에서는 은닉 상태와 현재 상태의 변화를 이후 상태에 전달하는 방법을 적용시키고자 한다 **??**.

이 장에서 개발할 충격파 이론 결합 모델은 모두 위에서 설명한 충격파의 개념을 적용시키고자 한다. 결합 모델은 입력 상태의 특정 변화를 어떤 식으로 감지하는지를 설계하고, 또 이 변화를 어떤 과정을 통해 미래 상태에 영향을 미치게 할 지를 설계하는 과정으로 구성되어 있다.

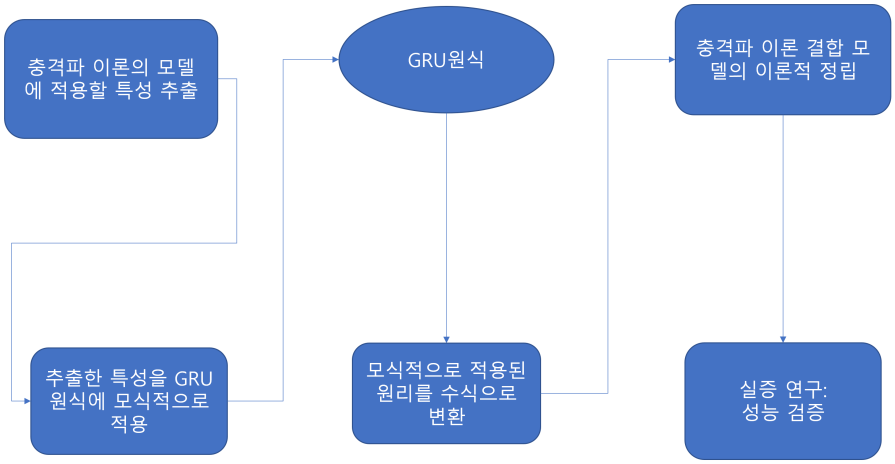

Figure 4.12: 충격파 이론 결합 모델 설계 방법론

입력 상태의 특정 변화는 각 지점에서 관측하며, 이에 따라 각 관측 지점에서 수집한 데이터 마다 충격파의 정도가 달라질 수 있다. 그러나 본 연구에서는 근접한 타 지점의 관측 충격파는 결합 모델에 반영하지 않았다. 이 부분은 먼저 기본적인 충격파 이론 결합 모델의 태동이 완료된 후에 후행되어야 할 것이다.

관측된 충격파는 충격파의 수치적 특성에 따라 달라지겠지만, 기본적으로 입력 상태의 Scale에 맞춰 일련의 Regularization과정을 수행해야한다. 그 뒤, 가공된 최종 충격파 수치는 입력 상태와 은닉 상태 행렬에 스칼라 곱으로 영향을 미칠 것이다. 본 장에서는 Regularization의 방법을 정립하는 것을 목표로 한다 **??**.

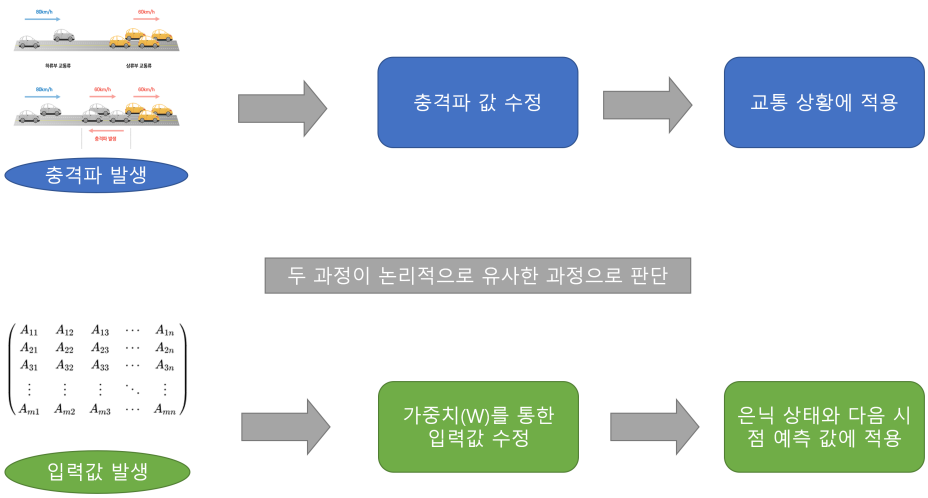

Figure 4.13: 충격파 이론과 GRU 모델의 공통점

4.1.2 충격파 속도의 사용

전통적인 충격파 이론, 혹은 운동파 이론에서는 충격파 속도를 수식으로 정의하고 있다.

$$w = \frac{\Delta Q}{\Delta \rho}$$

이 충격파 속도는 관측된 충격파에 수행할 Regularization에 응용할 수 있다.

먼저 위 식은 본 모델에서 요구하는 충격파의 양[17]이 아닌 입력값 속도(u)의 단위를 사용하는 충격파의 속도를 산출한다. 따라서 본 장에서는 결합 모델잇 설계를 통해 충격파 속도를 충격파 양으로 변환하는 과정을 서술한다. 또 본 연구에서 설계할 충격파 양 변환법이 모두 다른 Scale 데이터에도 일반적인 적용이 가능한지를 검증해야한다.

[17]충격파 양의 단위는 각 학습 단위에서의 데이터 Scale에 적응되어야 한다.

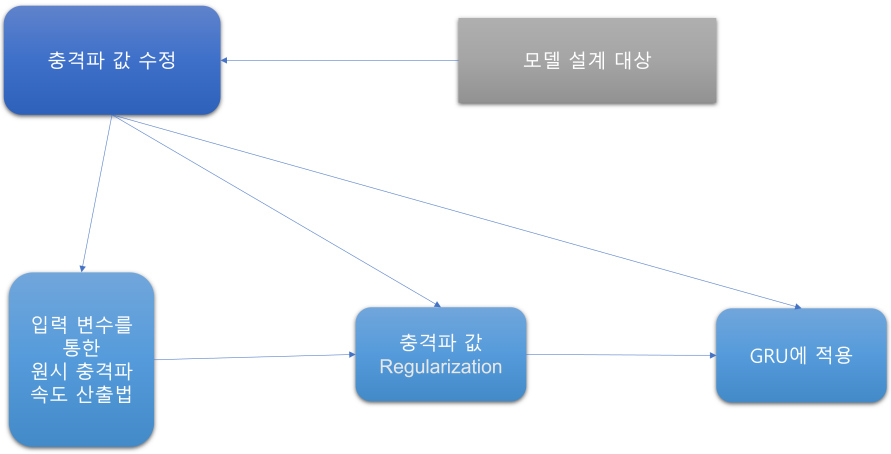

Figure 4.14: 모델 설계 대상

4.1.3 예측 결과 해석에 사용할 충격파 이론

본 연구에서는 교통 예측 모델을 단순히 수치적으로 해석하는 것이 아닌, 지리, 시각적 관점으로 해석할 것이다. 이때, 각 시점과 지점에서의 시각화 자료를 해석할 때, 현상 사이의 인과관계를 충격파 이론으로 해석할 수 있다.

4.2 충격파 이론 결합 모델의 모식적 설계 및 해석

수식적으로 충격파 결합 모델을 정립하는 것에 앞서, 먼저 모델의 전체적인 연산 과정을 모식적으로 구상하고, 논리적인 해석을 선행할 것이다. 또, 수식적으로 분석할 부분에 대한 고찰도 포함한다.

4.2.1 GRU에 대한 충격파 이론의 거시적 적용점

본 연구에서 개발할 결합 모델에서는 일련의 과정을 통해 산출된 충격파의 양을 적합한 적용점에 더하는 것으로 영향을 미칠 것이다. 이제부터 GRU모델의 각 과정에서의 충격파 양 적용점을 설정할 것이다.

GRU 모델은 기본적으로 은닉 상태와 입력 상태에 업데이트 게이트와 리셋 게이트를 거쳐 가중치와 편향의 영향을 미치게 하는 것으로 설명된다. 이때, 충격파 이론 결합 모델에서는 업데이트 게이트와 리셋 게이트, 그리고 추가로 후보 은닉 상태 수정 함수에 적용점을 설정할 것이다. 각 적용점에서는 산출된 충격파의 양이 동일하게 적용된다. 만약 Regularization과정이 올바르게 진행되지 않아 수치가 조금이라도 커지면 각 게이트에서의 영향이 너무 커져 예측 결과의 신뢰도가 떨어진다. 반대로 Regularization과정이 너무 약하게 진행되어 수치에 변동이 작아지면 각 게이트에 영향이 미미해져 기존의 GRU모델과의 차이점이 소실되어 버린다. 따라서 각 적용점의 수를 고려해 알맞은 Regularization과정을 탐색해야 한다. GRU의 리셋 게이트를 모식적으로 나타낸다면 다음과 같다 **??**.

먼저 은닉 상태와 입력 상태가 행렬로서 리셋 게이트를 거치게 된다. 리셋 게이트의 가중치 W_r이 각 행렬에 곱해진다. 그뒤 편향이 더해지고, 이것이 시그모이드 함수를 통해 0–1로 데이터가 축소된다. 이때, 편향이 더해지는 지점이 충격파 양을 더할 최적의 구간이라고 판단할 수 있다. 편향이 더해지는 구간은 지역 가중치가 적용된 데이터를 처리하기에 충격파 양을 더하여도 데이터에서 Regularization된 충격파의 영향이 확대되거나 소실되는 일이 일어날 수 없기 때문이다. 따라서 리셋 게이트의 편향을 더한 뒤, 거기에 충격파 양을 더하는 것으로 리셋 게이트에 영향을 줄 수 있다. 충격파가 강할수록 이전 상태의 정보를 덜 반영하도록 조정된다.

GRU의 업데이트 게이트는 구조적으로 리셋 게이트와 차이점이 한 점 없다. 따라서 적용점 또한 동일하며, 지역 가중치 W_z가 충격파 양에 의해 영향력이 줄어들 수

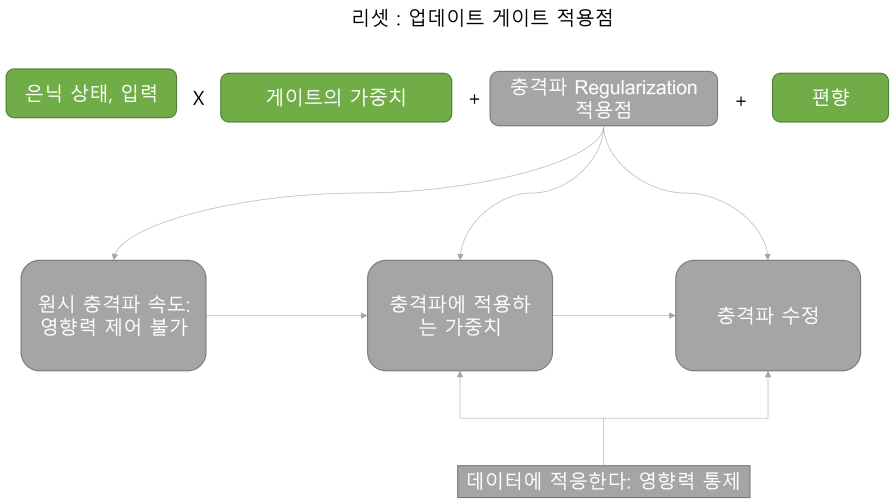

Figure 4.15: 적용점 모식도

있다. 충격파 강도에 따라 새로운 정보를 더 많이 반영할지, 기존 정보를 유지할지를 결정한다.

후보 은닉 상태는 최종적으로 산출된 은닉 상태와 입력 데이터에 마지막으로 충격파 양을 더한다. 이는 충격파 양의 확실한 영향을 위한 것이며, 영향에 따라 가중치 값이 조정되기에 예측 신뢰도를 유지할 수 있다.

4.2.2 충격파 속도의 산출

충격파 속도는 교통량과 속도의 변화를 기반으로 산출되는 원시 충격파 양이다. 충격파 속도는 산출 방법에 따라 실질 속도가 실수 값으로 산출될 수 있고, 충격파의 여부를 이진 분류하여 0, 1 Boolean[18] 값으로 산출하는 방법이 있다. 본 연구에서는 두가지 값을 모두 산출하는 방법을 사용하며, 실수 값으로 산출하는 방법과 이진

[18] 0과 1, True와 False 두 가지 값을 가지는 데이터 형식.

분류하는 방법 두개의 총 3가지의 충격파 이론 결합 모델을 개발한다. 충격파 속도 산출에 요구되는 유의 변수는 교통량과 속도이다.

가. LWR의 충격파 속도 산출법 적용

위에서 분석한 바와 같이 LWR에서의 충격파 속도는 입력 변수로 밀도와 교통량을 받는다. 그러나 본 연구에서의 요구 유의 변수는 밀도를 갖지 않는다. LWR의 충격파 속도에서는 밀도의 차이와 교통량의 차이를 통해 산출하지만, 본 연구에서의 충격파 결합 모델에서는 밀도 대신 교통량을 사용한다. 밀도는 단위구간 길이당 지나간 차량의 수이고, 교통량은 특정 지점을 지나간 차량의 수를 뜻한다. 밀도와 교통량은 차량의 밀집도를 측정하는 단위라는 공통점이 존재한다. 차량의 밀집도는 LWR의 충격파 속도에서 사용하는 밀도의 특성이기 때문에 본 연구에서는 밀도를 교통량으로 대체한 것이다. 또 기존 충격파 속도에서의 교통량은 속도로 변환하였다. LWR의 충격파 속도에서 사용하는 교통량의 특성은 차량의 밀집도가 아닌 차량의 흐름이다. 그리고 차량들의 평균 속도는 차량의 흐름을 반영하는 즉각적인 지표이다. 따라서 본 연구에서는 속도의 차량 흐름 특성을 교통량 대신에 사용하였다. 이 외에도, 기존 충격파 속도는 교통량과 밀도 사이의 관계인 미시적 지표와 거시적 지표의 변화 차이를 이용하여 산출한다. 본 연구에서 사용한 속도와 교통량 사의 관계도 전체적인 지표와 즉각적인 지표로 기존 충격파 속도와 동치를 보인다.

나. 표준편차의 임계값 사용

표준편차(standard deviation)는 평균에서 얼마나 멀리 떨어져 있냐를 나타낸 척도로, 관찰값들의 변동성을 나타내는 통계학적 척도다. 충격파는 기존 교통 흐름에 확연한 변화가 있을 때 발생한다. 이 때 확연한 변화의 기준에 대한 것은 자료의 기존 변동성을 벗어나는 관측값이 발생했을 때 충격파가 발생한다고 해석한다. 본 과정은 교통량의 변화값의 표준편차에 다른 수정을 가하지 않은 원시 표준편차를 임계값으

로 사용한다. 이 임계값을 기준으로 관측값의 변화율 절대값이 임계값 보다 작거나 같다면 0, 크다면 1로 산출한다 **??**.

일반적으로 표준편차는 모표준편차를 의미한다. 모표준편차(Population Standard Deviation)는 전체 모집단의 데이터 분포를 나타내는 통계적 지표로, 각 데이터가 평균으로부터 얼마나 떨어져 있는지를 측정한다. 이는 모집단 내 개별 데이터 값들이 평균값을 기준으로 어느 정도의 변동성을 가지는지를 정량적으로 표현하며, 모집단의 분포 특성을 파악하는 데 중요한 역할을 한다. 모표준편차는 모집단의 모든 구성원을 대상으로 계산되므로, 표본 표준편차와는 달리 표본의 편향을 수정할 필요가 없으며, 다음 수식에 의해 정의된다.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(X_i - \mu)^2}{N}}$$

여기서 σ는 모표준편차, N은 모집단의 크기, X_i는 각 데이터 값, μ는 모집단의 평균을 나타낸다. 모표준편차는 데이터 분포의 폭을 나타내며, 값이 클수록 데이터가 평균값을 기준으로 넓게 분포하고 있음을 의미하고, 값이 작을수록 데이터가 평균값에 더 가까이 모여 있음을 시사한다.

다. 이동평균의 임계값 사용

이동평균(Moving Average)은 시계열 데이터의 변동성을 완화하고 데이터의 추세를 파악하기 위해 사용되는 통계 기법이다. 이 방법은 일정한 구간 내에서 평균값을 계산하며, 새로운 데이터가 추가되면 가장 오래된 데이터를 제외하고 최신 데이터를 포함시켜 평균을 재계산한다. 일반적인 산술 평균과는 시간 개념의 유무에서 차이가 있다. 이동평균에는 단순이동평균, 누적이동평균, 가중이동평균 등의 종류가 존재하지만 본 연구에서는 가장 연산이 빠른 단순이동평균을 사용하였다.

특정 시간에서의 단순이동평균은 그 구간의 일반적인 교통 흐름을 나타낸다. 이

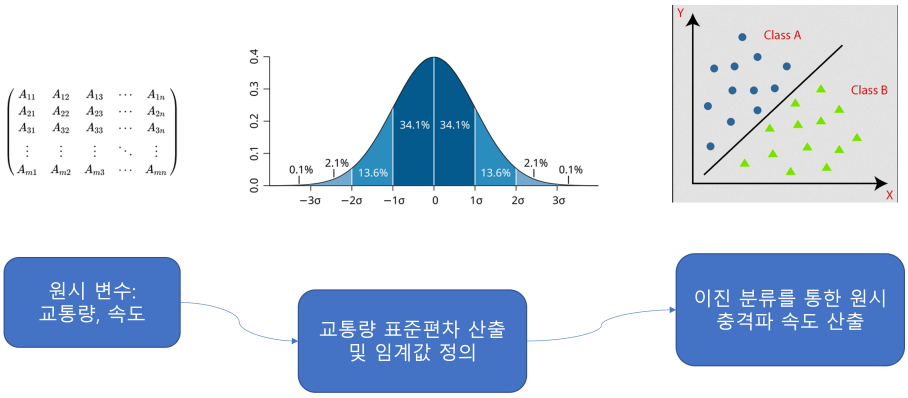

Figure 4.16: 표준편차의 임계값을 사용하는 충격파 속도 산출법 모식도

때, 이 교통 흐름인 이동평균의 표준편차를 구한다면, 기존 표준편차 기법보다 더욱 변동성을 정확하게 해석할 수 있다. 본 연구에서는 특정 시간 단위, 일명 window(창) N 값 만큼에 포함되는 교통량의 이동평균을 사용한다. 이는 적당한 관점(미시적, 거시적)에서의 교통 예측을 위해 고안하였다. 따라서 이 방법에서는 이동평균의 표준편차에 하이퍼파라미터 k를 곱한 값을 임계값으로 사용하여, 이 임계값을 기준으로 충격파 값을 이진 분류한다 **??**.

단순이동평균(Simple Moving Average, SMA)은 시계열 데이터의 변동성을 분석하기 위해 사용되는 기법으로, 일정 기간 동안의 데이터 값들의 산술 평균을 계산하여 추세를 파악한다. 이는 특정 기간 내의 데이터 값을 단순하게 더한 후 해당 기간의 데이터 개수로 나누며, 각 시점에서의 평균 값을 연속적으로 계산함으로써 데이터의 단기적 변동을 완화하고 추세를 시각적으로 나타낸다. 단순이동평균은 다음 수식에 의해 정의된다.

$$SMA_t = \frac{X_t + X_{t-1} + \cdots + X_{t-n+1}}{n}$$

여기서 SMA_t는 시점 t에서의 단순이동평균 값, X_t는 시점 t에서의 관측값, n은 이동평균을 계산하는 기간의 길이이다. 단순이동평균은 금융 데이터, 경제 지표, 생산량 등 다양한 분야에서 데이터의 단기적 변동을 분석하고 추세를 식별하는 데 사용된다.

Figure 4.17: 이동평균의 임계값을 사용하는 충격파 속도 산출법 모식도

라. ANN회귀 분석 사용

새로운 ANN회귀 모델을 만들어 충격파 속도를 예측 후, 예측 시점의 순환신경망 인풋에 병합하여 입력한다 **??**.

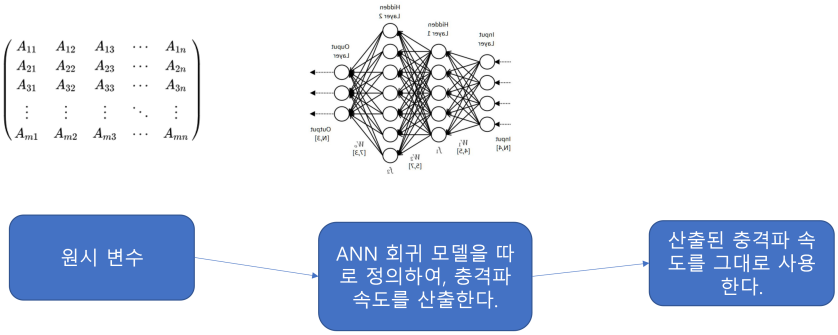

Figure 4.18: ANN 회귀모델을 사용하는 충격파 속도 산출법 모식도

4.2.3 충격파 Regularization

위 세가지 방법으로 구해진 원시 충격파 속도는 모델에 미칠 영향을 제어할 수 없어 예측 신뢰도가 0에 수렴한다. 따라서 산출된 충격파 속도에는 적합한 Regularization과정이 필수적이다. 이를 통해 충격파 양이 산출되고, 산출된 충격파 양은 모델에 적응하는 영향력을 가진다.

충격파 속도 Regularization의 목적인 데이터와 모델에 적응하는 영향력을 위해서 본 연구에서는 충격파 가중치를 고안하였다. 충격파 가중치는 각 모델에 추가된 충격파 양 산출을 위한 가중치로서, 원시 충격파 속도에 곱해짐으로서 이용된다. 다른 가중치와 동일하게 연산되며, 이 과정에서 손실 함수를 통해 모델과 데이터에 '적응하여' 충격파 양이 변동되게 된다. 이 충격파 가중치의 Shape은 [1, 유닛 수]이다. 따라서 각 유닛마다 적합한 충격파 가중치가 연산되게 된다.

충격파 가중치는 이진 분류된 충격파 속도와 실수 값의 충격파 속도 양 측에 같은 과정으로 연산된다. 위 세가지 과정에서 충격파 가중치는 모두 효용성을 지닌다.

4.3 충격파 이론 결합 모델의 수식적 설계

4.3.1 GRU에 대한 충격파 이론의 거시적 적용점의 수식

가중치까지 최종 산출된 충격파 양을 S라고 한다면, GRU에 각 적용점은 다음과 같이 표현된다.

먼저, 리셋 게이트의 적용점은 다음 수식으로 표현된다.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + S + b_r)$$

은닉 상태와 입력은 기본적인 GRU의 연산 과정인 가중치 곱셈을 수행한다. 그이후 가해지는 연산은 데이터를 축소하는 역할인 시그모이드 함수 이외에는 존재하지 않는다. 기존 GRU에서는 이 단계에서 편향을 사용했지만, 본 모델은 편향에 더해 영향을 줘야하는 충격파 양을 추가한다.

비슷한 형상으로 업데이트 게이트의 적용점은 다음 수식으로 표현된다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + S + b_z)$$

업데이트 게이트도 리셋 게이트와 마찬가지로 모든 데이터의 의미에 영향을 주는 연산이 다 끝난 뒤, 충격파 양을 편향과 함께 더한다.

마지막으로, 후보 은닉 상태와 은닉 상태는 다음 수식으로 표현된다.

$$\tilde{h}t = \tanh(W_h \cdot [r_t \odot ht - 1, x_t] + S + b_h)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

후보 은닉 상태는 후보 은닉 상태의 가중치가 모두 연산된 후, 편향과 함께 충격파 양을 더한다. 위 연산들과 같이 충격파 양은 총 3번 더해진다. 그러나 가중치로서 데이터에 적응되기에 그 영향력은 손실함수를 줄이는 방향으로 올바르게 학습될 수 있다.

거시적인 적용점은 위와 같고, 미시적인 충격파의 연산을 살펴보고자 한다.

4.3.2 충격파 속도의 산출

가. LWR모델의 충격파 속도 사용

LWR모델의 충격파 속도를 사용하는 충격파 속도 산출법은 다음과 같이 나타낼 수 있다.

$$w = \frac{\Delta Q}{\Delta p}$$

- w는 충격파의 속도
- Q는 교통량
- p는 시간평균속도

변수 내용은 모식적 설계에서와 같이 교통량과 속도를 사용하였으며, 그외 연산은 원형을 그대로 사용하였다.

나. 표준편차의 임계값 사용

표준편차의 임계값을 사용한 충격파 강도 산출은 다음과 같이 나타낼 수 있다.

$$\Delta Q_t = Q_t - Q_{t-1}$$

ΔQ_t는 현재 t 시점에서의 교통량 변화량을 뜻한다.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

- N은 데이터의 개수 (모집단의 크기)
- x_i는 i-번째 교통량 변화량
- μ는 모집단의 평균

본 연구에서는 모표준편차를 사용하였다. 물론, 모표준편차가 아닌 다른 종류의 표준편차를 사용하여도 무관하다. 그러나, 본 연구에서는 연산 속도의 감소도 모델 개발에 중요한 의미이기 때문에 연산 속도가 비교적 빠른 모표준편차를 사용하게 되었다.

$$s_t = \begin{cases} 1 & \text{if } |\Delta Q_t| > \sigma \\ 0 & \text{otherwise} \end{cases}$$

s_t는 충격파의 강도이다.

위에서 산출된 모표준편차 값을 임계값으로 그대로 사용한다. 임계값을 기준으로 0, 1 이진 분류한다.

다. 이동평균의 임계값 사용

이동평균의 임계값을 사용한 충격파 강도 산출은 다음과 같이 나타낼 수 있다.

$$\mathrm{MA}t = \frac{Q_t + Q_{t-1} + \cdots + Q_{t-(N-1)}}{N} = \frac{1}{N}\sum_{i=0}^{N-1} Q_{t-i}$$

Q_t는 t시점의 교통량이고, N은 하이퍼파라미터 Window(창)이다.

$$\Delta Q_t = Q_t - \mathrm{MA}_t$$

ΔQ_t는 교통량과 이동평균의 오차이다.

Window(창)은 각 시점에서 어느 정도의 영향력으로 임계값을 설정할 지 정의한다. 주식 시장의 차트 분석에 쓰이는 이동 평균선의 일 수와 동일한 설정이라 볼 수 있다. 이는 모델의 분석 목표가 거시적인 것인지, 미시적인 것인지에 따라 유동적으로 변경될 수 있다.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma = k\sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

- N은 데이터의 개수 (모집단의 크기)
- x_i는 i-번째 교통량과 이동평균의 오차
- μ는 모집단의 평균
- k는 임계값 조정을 위한 하이퍼파라미터
- σ는 임계값

단순 단순이동평균을 구하여 임계값을 사용한 것이 아닌, 이동평균의 표준편차를 구하여 임계값으로 활용하였다. 이때, K라는 하이퍼 파라미터를 추가로 곱하였다.

이는 표준편차값, 즉 임계값의 전반적인 추세를 선택할 수 있게 한다. 1로 설정하면 아무런 추세 반영이 없는, 기본적인 임계값이 된다. 만약 조금 낮게 설정한다면, 최근 추세 교통량이 낮다고 인식하게 된다. 반대로 높게 설정한다면, 최근 교통량이 높다는 추세를 가지게 된다. 이는 수동적인 데이터 분석이 뒷받침 될 것을 고려하였다. 그러나 이에 관한 성능 검정은 본 논문에서는 진행하지 않았다.

$$s_t = \begin{cases} 1 & \text{if } |\Delta Q_t| > \sigma \\ 0 & \text{otherwise} \end{cases}$$

교통량 변화율 ΔQ_t 의 절댓값이 임계값 σ 를 초과하면 $s_t = 1$, 그렇지 않으면 $s_t = 0$ 으로 설정한다.

4.3.3 충격파 Regularization

세가지 모델에 산출된 충격파 강도 s_t에 대해 공통적으로 다음과 같은 Regularization을 거친다.

$$W_{shockwave} = \begin{bmatrix} w_1 & w_2 & \dots & w_n \end{bmatrix}$$

$$S = W_{shockwave} \cdot s_t$$

$W_{shockwave}$는 충격파 가중치, S는 충격파 양이다. 충격파 가중치의 형상은 위 모식적 설계에서 정의한 것과 마찬가지로 1 by (데이터 수)의 형상을 띄고 있다. 즉, 순환 신경망에서는 유닛 수이다. 가중치의 연산은 일반적인 딥러닝 기법과 마찬가지로 원 데이터에 곱하는 방식으로 고안하였다.

4.4 충격파 이론 결합 모델의 이론적 정립

위 모델 설계를 종합하여 충격파 이론 결합 모델의 수식과 모식을 이론적으로 정립하였다.

4.4.1 ANN기반 충격파 이론 결합 모델

충격파 속도 산출을 위한 ANN모델은 다음으로 나타낸다.

$$\hat{y}_t = f(W \cdot X_t + b)$$

$\hat{y}_t$는 ANN모델의 t 시점 출력이다.

리셋 게이트는 다음과 같이 나타낸다. 변수들은 위의 설명과 동일하다.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + W_{shock} \cdot \hat{y}_t + b_r)$$

업데이트 게이트는 다음과 같이 나타낸다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + W_{shock} \cdot \hat{y}_t + b_z)$$

후보 은닉 상태는 다음과 같이 나타낸다.

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, x_t] + W_{shock} \cdot \hat{y}_t + b_h)$$

후보 은닉 상태는 리셋된 은닉 상태와 입력 데이터, 그리고 충격파 가중치와 강도를 결합하여 계산된다.

최종 은닉 상태는 다음과 같이 나타낸다.

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

최종 은닉 상태는 업데이트 게이트를 사용하여 이전 은닉 상태와 후보 은닉 상태를 조합된다. 여기서도 충격파의 영향을 반영하여 은닉 상태가 조정된다.

4.4.2 표준편차 기반 충격파 이론 결합 모델

교통량 변화율은 다음과 같이 나타낸다.

$$\Delta Q_t = Q_t - Q_{t-1}$$

여기서 Q_t는 시간 t 에서의 교통량(예: 차량 대수, 평균 속도 등)이다.

모표준편차를 이용한 임계값은 다음과 같이 나타낸다.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

- N은 데이터의 개수 (모집단의 크기)
- x_i는 i-번째 교통량 변화량
- μ는 모집단의 평균

최종 충격파 강도는 다음과 같이 산출된다.

$$s_t = \begin{cases} 1 & \text{if } |\Delta Q_t| > \sigma \\ 0 & \text{otherwise} \end{cases}$$

이제부터 실질 GRU 수식이다.

리셋 게이트는 다음과 같이 나타낸다. 변수들은 위의 설명과 동일하다.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + W_{shock} \cdot s_t + b_r)$$

업데이트 게이트는 다음과 같이 나타낸다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + W_{shock} \cdot s_t + b_z)$$

후보 은닉 상태는 다음과 같이 나타낸다.

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, x_t] + W_{shock} \cdot s_t + b_h)$$

후보 은닉 상태는 리셋된 은닉 상태와 입력 데이터, 그리고 충격파 가중치와 강도를 결합하여 계산된다.

최종 은닉 상태는 다음과 같이 나타낸다.

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

최종 은닉 상태는 업데이트 게이트를 사용하여 이전 은닉 상태와 후보 은닉 상태를 조합된다. 여기서도 충격파의 영향을 반영하여 은닉 상태가 조정된다.

4.4.3 이동평균 기반 충격파 이론 결합 모델

이동평균의 임계값을 사용한 충격파 강도 산출은 다음과 같이 나타낼 수 있다.

$$\mathrm{MA}t = \frac{Q_t + Q_{t-1} + \cdots + Q_{t-(N-1)}}{N} = \frac{1}{N}\sum_{i=0}^{N-1} Q_{t-i}$$

Q_t는 t시점의 교통량이고, N은 하이퍼파라미터 Window(창)

$$\Delta Q_t = Q_t - \mathrm{MA}_t$$

ΔQ_t는 교통량과 이동평균의 오차

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma = k\sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

- N은 데이터의 개수 (모집단의 크기)
- x_i는 i-번째 교통량과 이동평균의 오차
- μ는 모집단의 평균
- k는 임계값 조정을 위한 하이퍼파라미터
- σ는 임계값

$$s_t = \begin{cases} 1 & \text{if } |\Delta Q_t| > \sigma \\ 0 & \text{otherwise} \end{cases}$$

이제부터 실질 GRU 수식이다.
리셋 게이트는 다음과 같이 나타낸다. 변수들은 위의 설명과 동일하다.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + W_{shock} \cdot s_t + b_r)$$

업데이트 게이트는 다음과 같이 나타낸다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + W_{shock} \cdot s_t + b_z)$$

후보 은닉 상태는 다음과 같이 나타낸다.

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, x_t] + W_{shock} \cdot s_t + b_h)$$

후보 은닉 상태는 리셋된 은닉 상태와 입력 데이터, 그리고 충격파 가중치와 강도를 결합하여 계산된다.

최종 은닉 상태는 다음과 같이 나타낸다.

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

최종 은닉 상태는 업데이트 게이트를 사용하여 이전 은닉 상태와 후보 은닉 상태를 조합된다. 여기서도 충격파의 영향을 반영하여 은닉 상태가 조정된다.

4.4.4 LWR기반 충격파 이론 결합 모델

LWR모델의 충격파 속도를 사용하는 충격파 속도 산출은 다음과 같다.

$$w_t = \frac{\Delta Q}{\Delta p}$$

- w는 충격파의 속도
- Q는 교통량
- p는 시간평균속도

w_t는 t 시점의 충격파 속도이다.

이제부터 실질 GRU 수식이다.

리셋 게이트는 다음과 같이 나타낸다. 변수들은 위의 설명과 동일하다.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + W_{shock} \cdot w_t + b_r)$$

업데이트 게이트는 다음과 같이 나타낸다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + W_{shock} \cdot w_t + b_z)$$

후보 은닉 상태는 다음과 같이 나타낸다.

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, x_t] + W_{shock} \cdot w_t + b_h)$$

후보 은닉 상태는 리셋된 은닉 상태와 입력 데이터, 그리고 충격파 가중치와 강도를 결합하여 계산된다.

최종 은닉 상태는 다음과 같이 나타낸다.

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

최종 은닉 상태는 업데이트 게이트를 사용하여 이전 은닉 상태와 후보 은닉 상태를 조합된다. 여기서도 충격파의 영향을 반영하여 은닉 상태가 조정된다.

5장. 뉴웰의 차량추종모델 결합 모델 개발

5.1 뉴웰의 차량추종모델 특성 추출

뉴웰의 차량추종모델 모델은 미시적 관점의 교통류 이론으로서, 각 차량간의 차두 거리가 어떤 식으로 전파되는 지 설명하는 이론이다. 이 이론의 일부 특성을 추출하여 결합 모델을 설계할 것이며, 이를 모식적, 수식적으로 적용할 것이다. 또 뉴웰의 차량추종모델 모델은 충격파 이론의 응용이라 볼 수 있다. 따라서 충격파 이론과의 차이점도 중점적으로 분석한다 **??**.

5.1.1 뉴웰의 관성 모델의 개념 사용

뉴웰의 관성 모델은 차량 추종 모델로서, 차두 거리(헤드웨이)의 관계과 전파를 설명한다. 이 모델의 핵심은 반응 지연 시간(Reaction Delay), 속도 변화, 차량 간 간격을 기반으로 한 비선형적 관계를 설명하는 것이다. 이 이론또한 충격파 이론과 동일한 잣대에 의해 딥러닝 기법에 적용하기에 적합하다. 먼저, 뉴웰의 관성 모델은 차두 거리의 전파를 서술한다. 이는 가중치로서 설명될 수 있으며, 충격파 이론과 같이 일련의 과정으로 산출될 수 있다. 또 충격파 이론과의 차이점으로, 전파의 시간 지연이 중점적이다. 이는 각 데이터의 차두 거리 전파 스텝 수의 차이로서 적용하고자 한다.

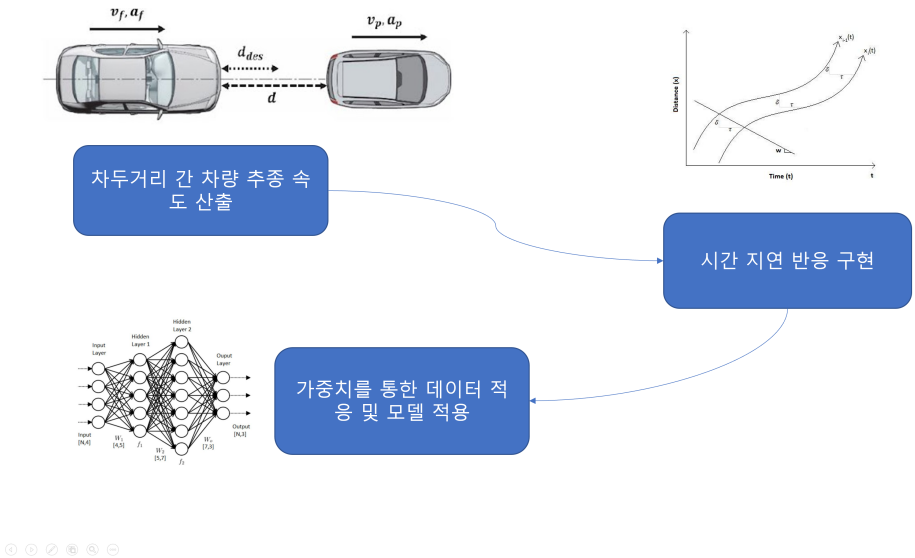

Figure 5.19: 뉴웰의 차량추종모델 설계 계획

5.1.2 비선형 속도 반응 함수 사용

뉴웰은 속도에 반응하는 차량 간의 관계를 다음 수식으로 나타냈다.

$$v_j(t) = G\left(z_{j-1}(t-\Delta) - z_j(t-\Delta)\right)$$

- $v_j(t)$는 시간 t에서 j번째 차량의 속도
- $z_{j-1}(t-\Delta)$는 앞차의 위치(시간 지연 Δ를 고려한)
- $z_j(t-\Delta)$는 j번째 차량의 위치(시간 지연을 고려한)
- $G(\cdot)$는 비선형 함수로, 속도 변화에 대한 운전자의 반응

이때, 본 연구에서는 $v_j(t)$를 가중치로서 합하여, 영향을 끼쳤다. 가중치로 합하는

부분은 충격파 이론의 결합과 마찬가지로 업데이트 게이트, 리셋 게이트, 후보 은닉 상태로 선정하였다. 단, 비선형 함수 G와 출력 형식은 GRU의 형식에 맞춰 변환하였다.

시간 지연에 대해서는 위의 속도 반응 출력 값을 특정 시간(스텝) 이후에 적용하는 방법으로 대체하였다. 이 방법을 통해 뉴웰 관성 모델의 개념적 원리는 모두 적용할 수 있다.

5.2 뉴웰의 관성 모델 결합 모델의 모식적 설계 및 해석

5.2.1 뉴웰의 관성 모델 결합 모델의 적용점

뉴웰의 관성 모델은 차량 추종 값을 통해 영향을 미친다. GRU 모델에서는 차량 추종 값의 영향을 충격파 이론과 마찬가지로 업데이트 게이트와 리셋 게이트, 후보 은닉 상태 게이트의 가중치 적용 후 편향 연산 과정에서 적용하는 것이 적합하다 판단하였다. 차량 추종 값이 연산 되었다면, 연산 된 값은 설계와 같이 특정 스텝 후의 시점 유닛에 적용점이 존재한다. 위와 같은 방법은 뉴웰의 관성 모델의 시간 지연과 차량 추종 원리를 GRU에 적용시킬 수 있을 것이다. 이처럼 본 연구에서 개발하고자 하는 뉴웰의 관성 모델 결합 모델의 적용점은 위에서 설계한 충격파 이론 결합 모델의 적용점과 동일하게 설정하였다. [19].

5.2.2 비선형 속도 반응 함수의 수정

본 연구에서는 비선형 속도 반응 과정을 매우 연산 친화적으로 수정하였다. 속도 반응 함수인 G는 추상적이고 연산 과정이 복잡해질 수 있는 가능성을 포함하고 있다. 이는 본 연구의 목적인 연산 속도 단축에 반대되며, 또 속도 반응 함수의 개발은 모델 연구의 복잡도만을 증가시키기에 본 연구에서는 속도 반응 함수 G를 완전히

[19]충격파 이론과의 특성이 일치하기 때문임

제거하였다. 즉, 속도 반응 값 $v_j(t)$를 시간 지연을 적용한 속도 차이로 설정하였다. 이를 통해 뉴웰의 차량 추종 이론의 이론적 목적은 GRU에 효과적으로 적용하면서, 연산 속도를 줄여 GRU의 장점 또한 보존해내었다 **??**.

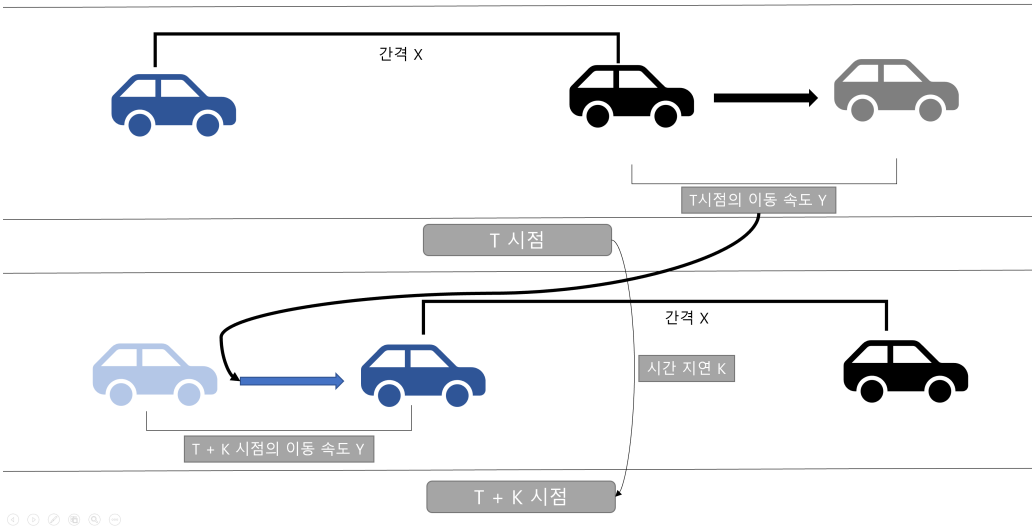

Figure 5.20: 뉴웰의 차량추종모델 적용 방법

위처럼 수정된 뉴웰의 비선형 속도 반응 함수를 GRU(Gated Recurrent Unit)에 적용함으로써, 교통류 모델링에서의 복잡한 비선형적 속도 변화를 정밀하게 처리할 수 있게 된다. 차량 속도 변화가 시간에 따라 은닉 상태의 갱신에 직접적으로 영향을 미치게 된다. GRU는 주로 시계열 데이터의 장기 의존성을 학습하는 데 효과적인 모델로, 기존에는 시간에 따른 단순한 패턴을 주로 학습해왔다. 그러나 뉴웰의 비선형 속도 반응 함수가 결합됨으로써 GRU는 차량 간의 비선형적인 속도 변화를 효과적으로 반영할 수 있게 된다. 차량 정체의 발생과 해소 등 미시적 교통 상황에 대한 모델의 예측 능력이 강화될 것이다. 예를 들어, 도로에서 앞차가 급감속하거나

정체가 발생할 경우, GRU는 뉴웰 모델을 통해 이러한 상황을 즉각적으로 반영하여 차량의 속도 변화를 비선형적으로 예측한다. 즉, 앞차와의 거리에 따라 변화하는 속도의 비선형적 패턴을 장기적으로 추적할 수 있다.

5.2.3 시간 지연의 적용

본 연구에서는 뉴웰의 관성 모델의 시간 지연 이론을 위에서 수정한 속도 반응 함수를 하이퍼파라미터인 스텝 수 k 스텝 뒤에 적용함으로써 적용하였다. 이를 GRU에 적용하면, 리셋 게이트와 업데이트 게이트에서 과거의 은닉 상태와 현재 입력 간의 관계를 학습하며, 차량의 반응이 즉각적이지 않고 지연될 수 있음을 모델에 반영할 수 있다. 시간 지연의 도입은 미시적 교통 흐름 예측에서 의미를 가진다. 차량이 앞차와의 간격을 조정할 때 발생하는 시간 지연은 혼잡 구간에서 정체를 유발하거나 해소하는 과정에 큰 영향을 미친다. 업데이트 게이트는 차량의 현재 상태를 얼마나 반영할지를 결정하는데, 시간 지연의 요소를 추가함으로써 장기적 의존성을 강화하여 교통 상황의 비선형적 변화를 더 잘 포착할 수 있을 것이다.

5.3 뉴웰의 관성 모델 결합 모델의 수식적 설계

5.3.1 비선형 속도 반응 함수

교통 데이터에서 속도 변화량을 계산하여 뉴웰의 관성 모델을 반영한다. 반응 시간이 지연된 입력 데이터를 예측에 사용하며, 반응 시간 k는 고정된 상수로 설정된다. 각 시점 t 에서 속도 변화량 G_t 는 다음과 같이 정의된다.

$$G(Q_t, Q_{t-1}) = Q_t - Q_{t-1}$$

G는 변화 속도를 산출하는 함수이다.

5.3.2 시간 지연

뉴웰의 관성 모델에 따라, k타임 스텝 뒤의 입력을 고려하여 시차 변화를 반영한 속도 변화량은 다음과 같이 표현된다.

$$G_{shifted} = G(Q_{t-k}, Q_{t-k-1})$$

k는 적용할 시간 지연 스텝 수이다.

5.3.3 GRU 적용

리셋 게이트는 다음과 같이 나타낸다. r_t는 이전 은닉 상태를 얼마나 "리셋"할지 결정하며, 이를 통해 새로운 입력 정보가 반영된다.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + G_{shock} \cdot G_{shifted} + b_r)$$

G_{shock}는 충격파 이론 결합 모델과 마찬가지로 영향력의 적응과 제어를 위한 가중치이다.

업데이트 게이트는 다음과 같이 나타낸다. z_t는 이전 은닉 상태와 현재 입력에 기반하여 은닉 상태를 얼마나 업데이트할지 결정한다.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + G_{shock} \cdot G_{shifted} + b_z)$$

후보 은닉 상태는 다음과 같이 나타낸다. $\tilde{h}_t$는 리셋된 이전 은닉 상태와 현재 입력을 사용하여 새로운 은닉 상태의 후보를 계산한다.

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, x_t] + G_{shock} \cdot G_{shifted} + b_h)$$

최종 은닉 상태는 다음과 같이 나타낸다. h_t는 업데이트 게이트를 통해 이전 은닉 상태와 새로운 은닉 상태 후보를 조합하여 최종 은닉 상태를 계산한다.

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

여기서 $\odot$ 는 원소별 곱(element-wise multiplication)을 의미하며, σ는 시그모이드 함수, tanh는 하이퍼볼릭 탄젠트 함수이다.

6장. 다상 교통 이론 결합 모델 개발

6.1 다상 교통 이론의 특성 추출

다상 교통 이론은 교통 상태를 여러 상황으로 나누어 다른 함수를 적용시키고, 상태간의 전이를 서술하는 이론이다. 본 연구의 다상 교통 이론 결합 모델은 3가지 상태를 사용하는 3상 교통 이론을 사용하였다.

다상 교통 이론의 개념 사용

다상 교통이론을 GRU 모델에 적용하는 개념은 복잡한 교통 흐름에서 발생하는 다양한 상태 변화를 정교하게 모델링하고 예측하는 것을 목표로 한다. 다상 교통이론은 교통 흐름을 여러 상(phase)으로 구분하며, 각 상은 교통 상태의 구체적인 특징을 반영한다. 본 연구에서는 다상 교통이론의 상전이를 딥러닝 모델과 결합 하기 위해 상태 전이 확률 행렬, 상태 영향 행렬, 상태 전이 영향 행렬의 세가지 하이퍼 파라미터를 고안하였다. 세가지 하이퍼 파라미터는 다상 교통이론의 상전이를 모델에 이식할 수 있다.

상태 전이 확률 행렬

상태 전이 확률 행렬은 교통 흐름의 현재 상태를 나타내는 확률 분포로, 다중 분류 모델의 출력에서 도출된다. 이 행렬은 소프트맥스 함수의 출력으로, 각 상태에 대한

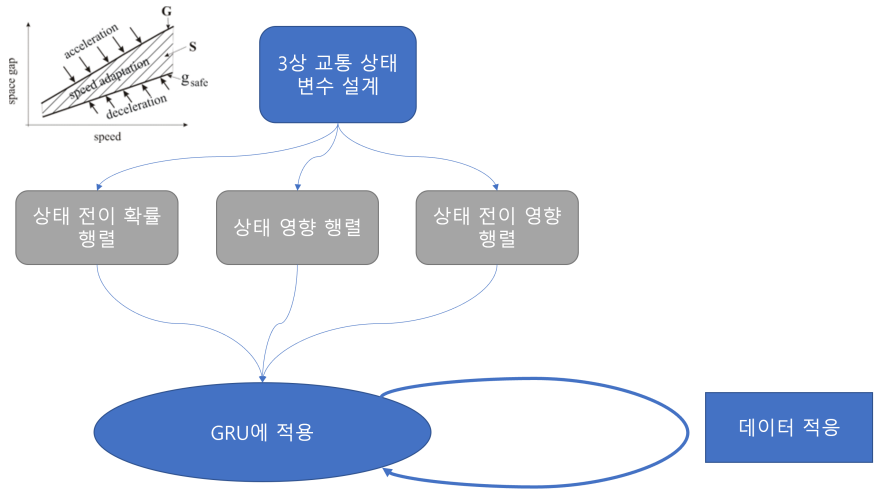

Figure 6.21: 다상 교통 이론 설계 방법

확률을 포함하며, 가장 높은 확률을 가진 상태가 현재 상태로 결정된다. 이 확률은 교통 흐름의 다양한 변동성과 불확실성을 반영하며, 현재 교통 상태를 실시간으로 모델링한다.

상태 영향 행렬

각 상태에서의 영향을 나타내는 하이퍼파라미터 행렬로, 각 교통 상태가 GRU 모델의 은닉 상태에 미치는 영향을 제어한다. 상태 영향 행렬은 특정 교통 상태에서 발생하는 교통 흐름의 특성과 해당 상태가 교통 예측에 미치는 영향을 반영하여, 모델이 교통 상황을 예측할 수 있도록 한다.

상태 전이 영향 행렬

상태 전이 영향 행렬은 교통 상태 전이가 GRU 모델의 은닉 상태에 미치는 영향을 나타낸다. 상태 전이는 혼잡 상태에서 자유 흐름 상태로의 전환, 혹은 그 반대의 전환이 발생할 때, 이 전이의 영향을 모델에 반영한다. 이 행렬은 특히 교통 혼잡이 발생하거나 해소되는 구간에서 역할을 다하며, 교통 흐름의 급격한 변화에 대한 반응을 모델에 포함시킨다.

6.1.1 다상 교통 이론의 상태전이 효과 적용

위에서 설명한 상태전이 하이퍼 파라미터를 GRU에 적용하는 방법을 살펴보고자 한다.

먼저, 전이 확률 행렬(S)을 통해 현재 상태에서 다음 상태로의 전이를 예측하는데, 이 행렬은 다중 분류 모델의 출력을 기반으로 하여 각 상태에 대한 확률을 계산한다. 이 확률 행렬은 각 교통 상태가 발생할 확률을 나타내며, 이를 통해 교통 흐름이 시간에 따라 어떻게 변화할지 예측할 수 있다. 이러한 확률 모델링은 교통류의 동적 특성을 반영하여 교통 상태 전이를 설명한다. 상태 전이는 교통 흐름의 급격한 변화를 반영하며, 특히 혼잡 상태로의 전이는 교통 정체가 발생하는 과정을 나타낸다. 예를 들어, 자유 흐름에서 혼잡 상태로의 전이는 차량 밀도의 급격한 증가와 함께 나타나며, 교통 상태 전이 확률은 이러한 상황에서 교통 혼잡의 형성 및 확산을 예측하는 역할을 한다. 상태 전이는 또한 교통 정체의 해소 과정에서 중요한 역할을 하며, 혼잡 상태에서 다시 자유 흐름으로의 전이는 차량 속도와 밀도의 회복을 반영한다.

상태 영향 행렬은 상태의 정적인 상태가 발생시키는 영향을 모델링한다. 예를 들어, 자유 흐름에서는 별다른 영향값이 존재하지 않지만, 혼잡 흐름에서는 차량의 속도를 감소시키려는 경향이 발생하게 된다.

상태 전이 영향 행렬(H)은 이러한 상태 전이가 GRU 모델의 은닉 상태에 미치는

영향을 반영한다. 이전 상태와 현재 상태 간의 전이는 교통 흐름에 큰 변화를 일으킬 수 있으며, 이 영향은 GRU 모델의 은닉 상태 업데이트에 반영된다. 예를 들어, 자유 흐름에서 혼잡 상태로의 전이는 차량의 속도와 흐름에 큰 변화를 일으키므로, GRU 모델은 이러한 전이에 따른 변화를 은닉 상태에서 학습하고, 이를 통해 교통 상태 변화를 예측할 수 있다.

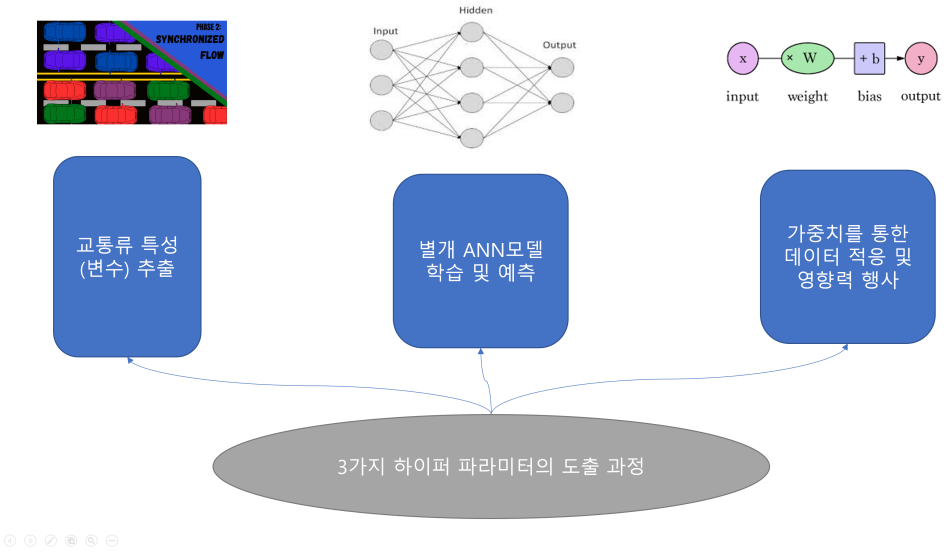

Figure 6.22: 다상 교통 이론 세가지 변수 추출 방법

6.2 다상 교통 이론 결합 모델의 모식적 설계 및 해석

다상 교통 이론 결합 모델의 모식적 설계 및 해석은 교통 흐름의 비선형적 특성을 정교하게 반영하고, 이를 통해 교통 상태 변화를 예측하는 접근법을 제시한다. 이 모델은 다상 교통 이론의 핵심 개념을 기반으로 교통 흐름을 여러 상태로 분류하고, 각 상태 간의 전이를 GRU(Gated Recurrent Unit)와 결합하여 동적 교통 흐름을

모델링한다.

6.2.1 상태 전이 확률 행렬(S)

상태 전이 확률 행렬은 S로 나타내며, 각 상태로의 전이될 확률을 나타낸다. 먼저, 상태 전이 확률 행렬(S)은 별개의 ANN 다중 분류 모델의 출력으로, 교통 흐름의 각 상태에 대한 확률을 나타내는 방법으로 고안하였다. 이는 소프트맥스 함수의 출력으로 얻어지며, 각 상태에 대한 확률 분포를 기반으로 가장 높은 확률을 가진 상태가 현재 교통 상태로 결정된다. 이 상태 전이 확률 행렬은 GRU 모델의 입력으로 활용되어 교통 상태의 변화를 반영하며, 각 상태가 GRU의 은닉 상태에 미치는 영향을 제어하는 역할을 한다.

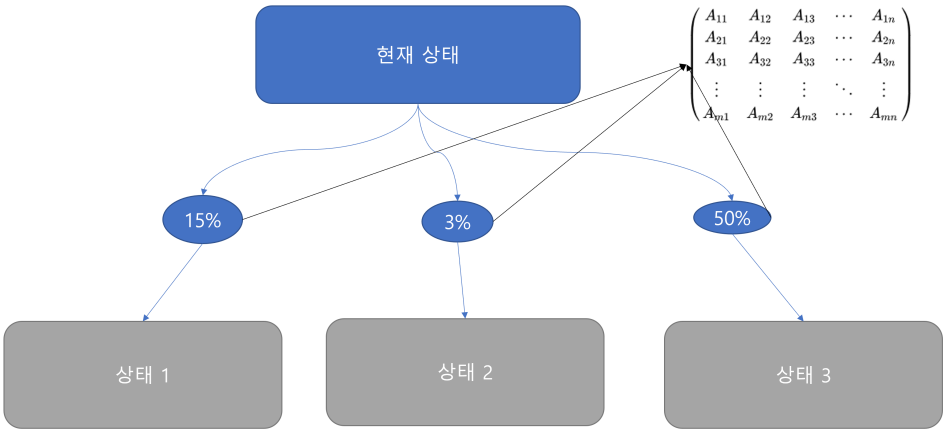

Figure 6.23: 상태 전이 확률 행렬 도식

6.2.2 상태 영향 행렬(P)

다음으로, 상태 영향 행렬(P)은 각 상태가 GRU의 은닉 상태에 미치는 영향을 나타내는 하이퍼파라미터 행렬이다. 각 상태는 고유한 Shape과 영향을 가지고 있으며, 해당 상태에서 발생하는 교통 흐름의 특성이 GRU의 은닉 상태 업데이트에 반영된다. 이를 통해 GRU 모델은 교통 흐름의 비선형적이고 복잡한 패턴을 학습할 수 있으며, 특정 상태에서 발생하는 교통 정체나 흐름 변화를 예측할 수 있다.

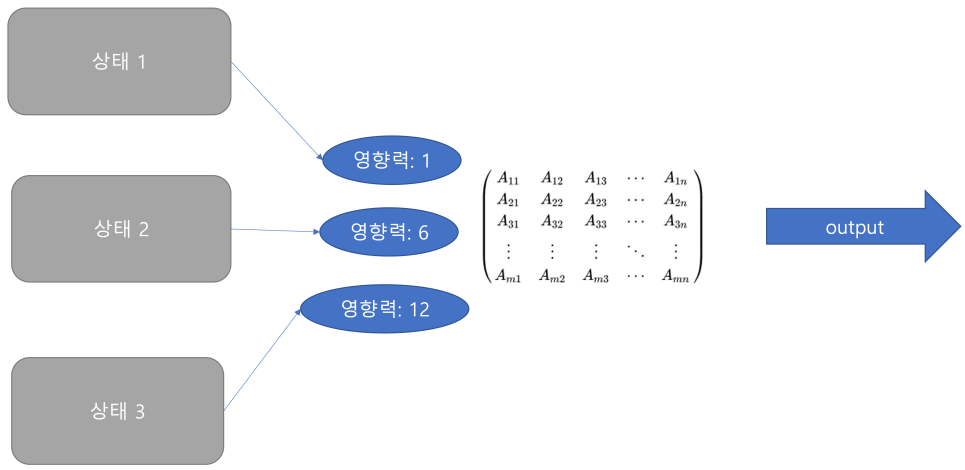

Figure 6.24: 상태 영향 행렬 도식

6.2.3 상태 전이 영향 행렬(H)

또한, 상태 전이 영향 행렬(H)은 상태 전이 과정에서 발생하는 영향을 반영하는 행렬로, 이전 상태와 현재 상태가 다를 경우 전이에 따른 영향을 GRU 모델의 은닉 상태에 반영한다. 이러한 상태 전이의 영향은 GRU의 은닉 상태 갱신에 반영되어,

교통 흐름이 시간에 따라 어떻게 변화하는지에 대한 예측이 가능하게 된다.

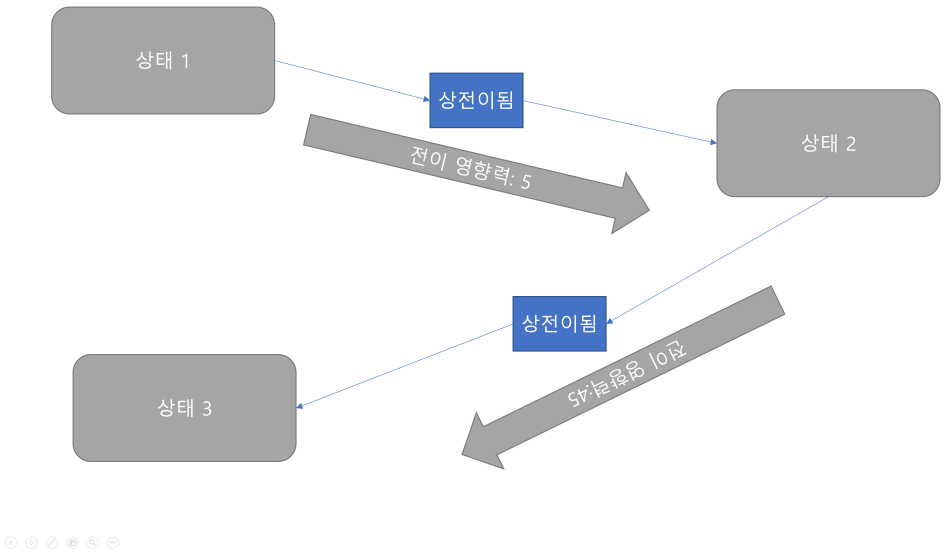

Figure 6.25: 상태 전이 영향 행렬 도식

6.2.4 전체 적용

이 결합 모델은 GRU의 시계열 처리 능력과 다상 교통 이론의 상태 전이 개념을 통합하여, 교통 흐름의 동적 변화를 반영한다. 교통 상태가 자유 흐름에서 혼잡 상태로 전이되면, 상태 전이 확률 행렬(S)은 해당 변화를 포착하고, 상태 영향 행렬(P)은 현재 상태가 교통 흐름에 미치는 영향을 반영하여 은닉 상태를 갱신한다. 이 과정에서 상태 전이 영향 행렬(H)은 이전 상태와 현재 상태 간의 전이를 고려하여 GRU 모델의 은닉 상태를 조정한다.

6.3 다상 교통 이론 결합 모델의 수식적 설계

다상 교통 이론을 GRU(Gated Recurrent Unit)와 결합한 모델의 수식적 설계는 교통 흐름의 상태 전이와 영향을 은닉 상태에 반영하여 교통 예측의 정확도를 높이는 것을 목표로 한다. 이를 위해 각 상태의 전이 확률과 영향치를 수식적으로 표현하고, GRU의 게이트 구조에 통합하여 설명한다.

6.3.1 상태 전이 확률 행렬 S

상태 전이 확률 행렬 S는 다중 분류 모델을 통해 얻어진다. 각 상태 i에서 j로의 전이는 다음과 같이 정의된다.

$$S_{ij} = P(\text{State}_j | \text{State}_i)$$

여기서 S_{ij}는 현재 상태 i에서 다음 상태 j로의 전이 확률이다. 이 확률은 소프트맥스 함수로 도출되며, 다음과 같이 정의된다.

$$S_{ij} = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

여기서 z_j는 상태 j에 대한 활성화 값이며, k는 가능한 모든 상태를 나타낸다.

6.3.2 상태 영향 행렬 P

상태 영향 행렬 P는 각 상태에서 GRU 은닉 상태에 미치는 영향을 나타낸다. 상태 i의 영향치는 P_i로 나타내며, 이는 은닉 상태에 다음과 같이 더해진다.

$$h_t = h_t + P_i$$

이때 P_i는 현재 상태에 해당하는 영향치를 나타내는 하이퍼파라미터이다. P는 모델 학습 중에 조정되며, 특정 상태에서의 교통 흐름 변화가 은닉 상태 갱신에 어떻게 영향을 미치는지를 제어한다.

6.3.3 상태 전이 영향 행렬 H

상태 전이 영향 행렬 H는 상태 전이가 발생할 때 그 전이가 은닉 상태에 미치는 영향을 나타낸다. 상태 i에서 상태 j로의 전이에 따른 영향을 H_{ij}로 나타낸다. 이는 은닉 상태에 다음과 같이 반영된다.

$$h_t = h_t + H_{ij}$$

이때, H_{ij}는 상태 i에서 상태 j로 전이할 때 발생하는 영향치로, 교통 상태 전이가 은닉 상태에 미치는 변화를 설명한다.

6.4 다상 교통 이론 결합 모델의 이론적 정립

다음은 위에서 제시된 다상 교통 이론 결합 모델의 수학적 표현을 기술한 내용이다.

6.4.1 상태 전이 확률 계산

주어진 입력 $\mathbf{x}_t$에 대해 상태 전이 확률 행렬 $\mathbf{S}_t$는 Softmax 함수를 통해 도출된다. 이는 현재 시점에서의 상태가 가질 확률 분포를 나타낸다. 상태 전이 확률 행렬은 다음과 같이 정의된다.

$$\mathbf{S}_t = \text{softmax}(\mathbf{x}_t)$$

본 연구에서 사용된 3상 이론의 경우, 상이 3개이기 때문에 다음과 같이 3×3의 shape을 띈다.

$$\mathbf{S}_t = \begin{pmatrix} 0.1 & 0.2 & 0.1 \\ 0.2 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix}$$

6.4.2 상태 영향 행렬 계산

상태 영향 행렬(P)은 하이퍼 파라미터로, 적절한 값을 모델마다 설정해주어야 한다. 본 연구에서는 상태 전이 영향과 마찬가지로 별개 회귀 모델을 사용하여 설정하였다. 3상 이론이라면, 경우의 수에 따라 다음과 같은 3×3 shape을 가진다.

$$\mathbf{P} = \begin{pmatrix} 2 & 3 & 1 \end{pmatrix}$$

6.4.3 현재 상태 결정

상태 전이 확률 행렬 $\mathbf{S}_t$에서 가장 높은 확률을 갖는 상태를 현재 시점 t의 상태로 선택하며, 이 상태는 다음과 같이 정의된다.

$$s_t = \arg\max(\mathbf{S}_t)$$

여기서 s_t는 현재 시점에서 선택된 상태를 의미한다.

6.4.4 이전 상태와 상태 전이 영향 계산

이전 시점 $t-1$의 은닉 상태 $\mathbf{h}_{t-1}$에 대해 Softmax 함수를 적용하여 이전 상태 s_{t-1}를 결정한다.

$$s_{t-1} = \arg\max(\text{softmax}(\mathbf{h}_{t-1}))$$

이전 상태 s_{t-1}와 현재 상태 s_t에 대해 상태 전이 행렬 $\mathbf{H}$에서 상태 전이 영향을 다음과 같이 계산한다.

$$\mathbf{H}_{s_{t-1},s_t} = \mathbf{H}[s_{t-1}, s_t]$$

6.4.5 현재 상태 영향 계산

현재 상태 영향은 설정된 상태 영향 행렬(P)의 값 중 위에서 도출된 현재 상태 s_t에 대한 값을 사용한다. 현재 상태 영향 p_t는 다음과 같이 연산된다.

$$p_t = P[s_t]$$

6.4.6 실질 GRU 연산

기본 GRU의 동작은 게이트 메커니즘을 통해 상태를 업데이트하며, 이를 다음과 같이 표현할 수 있다.

업데이트 게이트 $\mathbf{z}_t$와 리셋 게이트 $\mathbf{r}_t$는 다음과 같이 계산한다.

$$\mathbf{z}_t = \sigma(\mathbf{W}_z\mathbf{x}_t + \mathbf{U}_z\mathbf{h}_{t-1} + \mathbf{H}_{s_{t-1},s_t} + p_t + \mathbf{b}_z)$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r\mathbf{x}_t + \mathbf{U}_r\mathbf{h}_{t-1} + \mathbf{H}_{s_{t-1},s_t} + p_t + \mathbf{b}_r)$$

새로운 은닉 상태 후보 $\tilde{\mathbf{h}}_t$는 다음과 같다.

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{H}_{s_{t-1},s_t} + p_t + \mathbf{b}_h)$$

최종 은닉 상태 $\mathbf{h}_t$는 다음과 같다.

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t$$

6.4.7 상태 영향 및 상태 전이 영향 반영

위의 기본 GRU계산에 본 모델의 핵심인 영향 값들을 반영하는 방법은 다음과 같다. 위에서 고안한 것과 같이 상태 영향 벡터 $\mathbf{P}$에서 현재 상태 s_t에 해당하는 값을 $\mathbf{P}_{s_t}$로 정의한다. 이는 상태 s_t에 해당하는 상태 영향을 나타낸다.

$$\mathbf{P}_{s_t} = \mathbf{P}[s_t]$$

이전 시점의 상태 s_{t-1}와 현재 상태 s_t 간의 상태 전이 영향 $\mathbf{H}_{s_{t-1},s_t}$와 상태 영향 $\mathbf{P}_{s_t}$를 은닉 상태 $\mathbf{h}_t$에 반영하여 최종 은닉 상태를 업데이트한다.

$$\mathbf{h}_t^{\text{updated}} = \mathbf{h}_t + \mathbf{P}_{s_t} + \mathbf{H}_{s_{t-1},s_t}$$

이렇게 산출된 은닉 상태는 위 기본 GRU계산에 적용되어 다상 교통 이론의 특성을 가지게 된다. 여기서,

- $\mathbf{x}_t$: 현재 시점 t에서의 입력 벡터
- $\mathbf{h}_{t-1}$: 이전 시점 $t-1$의 은닉 상태
- $\mathbf{P}$: 상태 영향 행렬

- $\mathbf{H}$: 상태 전이 영향 행렬
- $\mathbf{S}_t$: 현재 시점에서의 상태 전이 확률 행렬 (Softmax 함수를 사용하여 계산)

7장. 데이터 구성

본 연구에서는 위에서 설정한 요구 유의 변수를 충족하는 실질 데이터를 수집하였다. 교통사고 데이터, 돌발 정보, 정체 및 혼잡, 소통 정보, 기상 정보 등의 데이터를 수집하였다. 당일 단기 교통량 분석과 기상에 따른 교통량 분석을 목표로 하기에 각 분기별 1개월씩 1월, 4월, 7월, 10월 총 4개월 데이터를 수집하였다. 각 분기의 첫 달을 수집하여 계절별 기상의 특성을 뚜렷히 반영하고, 데이터 과적합 문제를 해결하고자 한다. 데이터는 국가교통정보센터, 한국도로공사, 기상청 등의 공신력있는 공공기관에서 수집하였다. 또한 각 데이터들은 데이터 전처리 과정을 거쳐 모델의 실질적 입력 데이터로 변환하였다.

7.1 데이터 구성 과정

본 연구에서 사용된 데이터는 모두 파일 데이터 형식으로 수집되었는데, 다음과 같은 과정을 통해 수집되었다. 먼저, 데이터 제공기관(한국 도로 공사, ITS 국가교통정보센터 등)에서 원시 파일 데이터를 다운로드 받는다(zip, csv 등). 원시 파일 데이터는 분석 개발 프로그래밍 언어인 Python을 사용하여 처리되었으며, 특히 데이터 처리 패키지인 Pandas를 사용하여 데이터를 처리하였다. ITS 국가교통정보센터의 교통소통정보 데이터처럼 개별 파일 당 크기가 1GB를 넘어가는 대용량 데이터의 경우, 병렬 처리 연산을 지원하는 Dask 패키지를 사용하였다.

7.2 데이터 수집

데이터 템플릿에서 설정한 요구 유의 변수를 만족하는 구성의 데이터들을 수집하였다. 모든 데이터는 파일 데이터 형식으로 수집되었다.

7.2.1 교통 직접 변수

교통 직접 변수 혼잡빈도, 교통량과 속도를 수집하였다. 혼잡빈도와 교통량 데이터는 한국도로공사의 혼잡빈도 데이터를 수집하였고 **??**, 속도 데이터는 ITS 국가교통정보센터의 교통 소통 정보 데이터를 수집하였다 **??**. 한국도로공사의 혼잡빈도 데이터는 교통량과 속도, 혼잡빈도와 차로 번호가 VDS 코드를 지점으로 하여 구성되어 있다. ITS 국가교통정보센터는 5분 단위로 교통 소통 정보 데이터는 속도와 표준노드링크 데이터로 구성되어 있다. 추가로 요구 유의 변수에는 존재하지 않지만, 교통류에 직접적으로 영향을 미치는 돌발 정보 데이터도 수집하였다 **??**. 돌발 정보 데이터는 ITS 국가교통정보센터의 교통 돌발 정보 데이터를 수집하였다. ITS 국가교통정보센터의 교통 돌발 정보 데이터는 충격파라고 할 수 있는 정보를 원시적으로 담고 있다. ITS 국가교통정보센터의 돌발정보 데이터는 돌발 일시와 돌발 구분[20], 위, 경도 데이터 등으로 구성되어 있다. 단기간 교통사고 정보가 필요하기에 도로교통공단 월별 요일별 교통사고통계 데이터를 파일데이터 형태로 수집하였다. 그러나 교통사고 정보는 데이터의 형태가 활용하기 적합하지 않아 수집한 뒤 학습에는 사용하지 않았다.

[20]돌발 구분, 돌발 상세 구분, 돌발 내용으로 구성되어 있다.

11 rows 1616869 rows × 6 columns

	생성일	생성시분	링크ID	도로관리기관	통행속도	통행시간
0	20230101	2345	2510159608	NaN	26	0
1	20230101	2345	2510159700	NaN	25.5	0
2	20230101	2345	2510159800	NaN	21	0
3	20230101	2345	2510159900	NaN	31	0
4	20230101	2345	2510159901	NaN	20.5	0
...	...	...	...	...	...	...
1616864	20230101	2355	4180383901	NaN	4	0
1616865	20230101	2355	4180384001	NaN	97	0
1616866	20230101	2355	4180384101	NaN	76	0
1616867	20230101	2355	4180384201	NaN	115	0
1616868	20230101	2355	4180384301	NaN	79	0

Table 7.13: 교통소통정보

11 rows 1131 rows × 13 columns

	돌발일시	관리기관	도로타입	도로번호	도로명	방향	돌발구분	돌발상세구분	돌발내용	X	Y	링크아이디	돌발종료일시
0	2023-01-01 13:12:27	한국도로공사	고속국도	60	서울양양선	양양 방향	기타돌발	고장	(갓길)승용차고장처리중	128.4929	37.9769	2650168000	-
1	2023-01-01 12:41:17	한국도로공사	고속국도	45	중부내륙선	창원 방향	기타돌발	차량증기/정체	차량증기/정체	127.6062	37.2134	2370601900	-
2	2023-01-01 11:05:13	한국도로공사	고속국도	65	동해선(부산-포1	포항 방향	교통사고	사고	(갓길)승용차관련사고처리중	129.4162	35.7734	3520264000	-
3	2023-01-01 15:48:21	한국도로공사	고속국도	40	평택제천선	제천 방향	기타돌발	차량증기/정체	차량증기/정체	127.6166	36.9861	2010051300	-
4	2023-01-01 14:13:00	한국도로공사	고속국도	25	논산천안선	천안 방향	기타돌발	차량증기/정체	-차량터널(263K)3km정체	127.1026	36.0819	-	-
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1126	2023-01-01 15:04:00	경찰청	시군도	0	가양대로	-	공사	-	<공사> 가양로 구룡사거리 마포구고양시덕양구 1차로 주의운전 도로공사	126.872128	37.581869	2180147300	2023-01-01 18:00:00
1127	2023-01-01 15:12:00	경찰청	일반국도	38	서동대로	-	공사	-	<공사> 일반국도38호선 안양교량교차로 내리사거리 2차로 주의운전 도로	127.215717	37.809286	2310014402	2023-01-01 17:00:00
1128	2023-01-01 15:22:00	경찰청	특별광역시도	0	계백로1716번길	-	공사	-	<공사> 계백로1716번길 센트럴파크3차 금잣초등학교사거리 1차로 주의운	127.411943	36.319642	1840126300	2023-01-01 18:00:00
1129	2023-01-01 16:19:00	경찰청	특별광역시도	0	국회대로	-	교통사고	-	<사고> 국회대로 여의2교남단교차로 의원회관앞교차로 4차로 주의운전 :	126.912093	37.526136	1180031502	2023-01-01 16:30:00
1130	2023-01-01 18:56:00	경찰청	고속국도	30	당진영덕고속도로	-	교통사고	-	<사고> 청주영덕고속도로 안동시의성군 안동JC 2차로 156 Km (안동JC	128.630014	36.431557	3540811600	2023-01-01 19:26:00

Table 7.14: 교통돌발정보

5 rows × 13 columns

	집계년월	집계시분	VDS_ID	기점종점방향구분코드	노선번호	지점이정	도로이정	평균교통량	평균속도	혼잡빈도수	차로번호	콘존명	콘존길이
0	202301	01:50	0100VDS13000	S	100	154.4	156.0	7	91.17	0	2	동김해IC-서김해IC	4.5
1	202301	01:50	0100VDS13000	S	100	154.4	156.0	5	75.26	0	3	동김해IC-서김해IC	4.5
2	202301	01:50	0100VDS13100	S	100	155.4	156.0	3	113.83	0	1	동김해IC-서김해IC	4.5
3	202301	01:50	0100VDS13100	S	100	155.4	156.0	7	94.57	0	2	동김해IC-서김해IC	4.5
4	202301	01:50	0100VDS13100	S	100	155.4	156.0	4	80.61	0	3	동김해IC-서김해IC	4.5

Table 7.15: 혼잡빈도

7.2.2 도로환경변수

도로환경변수는 모두 ITS 국가교통정보센터의 표준노드링크 데이터를 사용하였다 **??**, **??**. 국토교통부 도로 현황 데이터 또한 결측지 보정을 위해 대안으로서 수집하였다 **??**. 표준노드링크 데이터는 요구 유의 변수인 출발 지점, 끝 지점 정보와 도로 폭, 차로 개수, 도로 위치가 포함된 shp파일로 되어있다. 좌표계는 epsg:4326을 사용하여 위, 경도 데이터를 사용하였다. 도로 길이 또한 데이터 템플릿과 마찬가지로 하버사인 함수에 대입하여 산출하였다.

11 rows 206116 rows × 9 columns Static Output

	NODE_ID	NODE_TYPE	NODE_NAME	TURN_P	UPDATEDATE	REMARK	HIST_TYPE	HISTREMARK	geometry
0	3698003600	101	¿Ô°ÜÃx³ãÉ¸Ã¸CXÕ	1	20230519	NaN	NaN	NaN	POINT (128.48143291923889 35.98756183745635)
1	3698004304	101	¿Ô°ÜÃBXÕÃÉµIÇÒ±³	1	20230519	NaN	NaN	NaN	POINT (128.40517889519722 35.99951853267461)
2	3698004601	101	X®ÃÜ¸®BB4-7	1	20230519	NaN	NaN	NaN	POINT (128.39722570374218 36.00145561148545 4)
3	3698005000	103	XÓX°'°ÊSHYÅ¡	0	20230519	NaN	NaN	NaN	POINT (128.39197629411416 36.00232749395027 6)
4	3698011100	101	ÇØXÕX1¸°ÃÎÅý	0	20230519	NaN	NaN	NaN	POINT (128.39735795896786 36.07650990055222)
...	...	...	... ...	...	...	...	...	... ...	
206111	3688175100	101	¿³»Ã,®927	0	20230519	NaN	NaN	NaN	POINT (128.28431169045686 35.92450917887177)
206112	3688179500	101	´®»Ã,®1D44	0	20230519	NaN	NaN	NaN	POINT (128.32688295624982 35.92389285701867)
206113	3688181600	102	ÃÜÃ´,®»E190-1	0	20230519	NaN	NaN	NaN	POINT (128.2247550524357 35.85974943677927 5)
206114	3698001100	103	XÓX°'°ÊSHYÅ¡	0	20230519	NaN	NaN	NaN	POINT (128.34763407601076 35.96086871244753 5)
206115	3698001500	101	¿µ,®1D84	0	20230519	NaN	NaN	NaN	POINT (128.35139335612965 35.96568230647233)

Table 7.16: 노드 데이터

11 rows 548593 rows × 22 columns Static Output

	LINK_ID	F_NODE	T_NODE	LANES	ROAD_RANK	ROAD_TYPE	ROAD_NO	ROAD_NAME	ROAD_USE	MULTI_LINK	...	REST_VEH	REST_W	REST_H	C-ITS	LENGTH	UPDATEDATE	REMARK	HIST_TYPE
0	2710802605	2710246080	2710245808	2	101	0	35	ÃB°Ì°ìXÕµµ·Ì	0	0	...	5	0	0	0	802.513275	20240223	NaN	LINK1007
1	2710802506	2710246380	2710008408	2	101	0	35	ÃB°Ì°ìXÕµµ·Ì	0	0	...	5	0	0	0	911.416276	20240223	NaN	LINK1007
2	2710802607	2710246280	2710246008	2	101	0	35	ÃB°Ì°ìXÕµµ·Ì	0	0	...	5	0	0	0	918.955738	20240223	NaN	LINK1007
3	2710706908	2710245080	2710244608	2	107	0	-	-	0	0	...	0	0	0	0	59.268985	20240223	NaN	LINK0001
4	2150268601	2150102880	2150102908	1	107	0	-	-	0	0	...	0	0	0	0	7.638523	20231211	NaN	LINK0005
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
548588	2870019411	2870008101	2870078408	2	101	0	15	XÕHYÇØXÊ°ìXÕµµ··	0	0	...	5	0	0	0	11066.335812	20230519	NaN	NaN
548589	3130000103	3130000101	3130000104	1	106	0	742	³éÃà·Ì	0	0	...	0	0	0	0	5620.608180	20230519	NaN	NaN
548590	3510029708	3510012607	3510015403	1	105	0	20	ÇØXÉ·Ì	0	0	...	0	0	0	0	2814.570321	20230519	NaN	NaN
548591	1860014904	1860006902	1860007800	1	104	0	-	±Ý³²±¸ÃÌ·Ì	0	0	...	0	0	0	0	1708.927407	20230519	NaN	NaN
548592	2650003820	2650001502	2650001501	1	106	0	446	±è°Ì´®¿Ô·Ì	0	0	...	0	0	0	0	1127.902246	20230519	NaN	NaN

11 rows 548593 rows × 22 columns Static Output

	D_TYPE	ROAD_NO	ROAD_NAME	ROAD_USE	MULTI_LINK	...	REST_VEH	REST_W	REST_H	C-ITS	LENGTH	UPDATEDATE	REMARK	HIST_TYPE	HISTREMARK	geometry
	0	35	ÃB°Ì°ìXÕµµ·Ì	0	0	...	5	0	0	0	802.513275	20240223	NaN	LINK1007	2.710803e+09	LINESTRING (127.41888724096204 36.63414446043 23...
	0	35	ÃB°Ì°ìXÕµµ·Ì	0	0	...	5	0	0	0	911.416276	20240223	NaN	LINK1007	2.710803e+09	LINESTRING (127.41812774275179 36.64233159867 3...
	0	35	ÃB°Ì°ìXÕµµ·Ì	0	0	...	5	0	0	0	918.955738	20240223	NaN	LINK1007	2.710803e+09	LINESTRING (127.41864389726209 36.64233885481 8...
	0	-	-	0	0	...	0	0	0	0	59.268985	20240223	NaN	LINK0001	NaN	LINESTRING (127.42113186443512 36.61694212815 67...
	0	-	-	0	0	...	0	0	0	0	7.638523	20231211	NaN	LINK0005	NaN	LINESTRING (127.05819075525902 37.96490493535 6...
	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
	0	15	XÕHYÇØXÊ°ìXÕµµ··	0	0	...	5	0	0	0	11066.335812	20230519	NaN	NaN	NaN	LINESTRING (126.55638389157494 36.37708488679 21...
	0	742	³éÃà·Ì	0	0	...	0	0	0	0	5620.608180	20230519	NaN	NaN	NaN	LINESTRING (127.44496109961187 35.44361028763 8...
	0	20	ÇØXÉ·Ì	0	0	...	0	0	0	0	2814.570321	20230519	NaN	NaN	NaN	LINESTRING (129.39739718553136 36.16161397033 6...
	0	-	±Ý³²±¸ÃÌ·Ì	0	0	...	0	0	0	0	1708.927407	20230519	NaN	NaN	NaN	LINESTRING (127.38433367311349 36.44783871203 8...
	0	446	±è°Ì´®¿Ô·Ì	0	0	...	0	0	0	0	1127.902246	20230519	NaN	NaN	NaN	LINESTRING (128.15686287581545 37.91387988602 6...

Table 7.17: 링크 데이터

11 rows 260549 rows × 19 columns Static Output

	PRESENT_SN	LCLAS_CL	MLSFC_CL	SCLAS_CL	ATRB_SE	WTNNC_SN	NTFC_SN	DGM_NM	DGM_AR	DGM_LT	SIGNGU_SE	GRAD_SE	ROAD_TY	ROAD_NO	ROAD_ROLE	DRAWING
0	44790UQ151P5202002170001	UQS120	NaN	NaN	UQS120	44790URZ202002170001	44790NTC202002170001	소로1류	2866.0	516.0	44790	소로	1	1	PMI0004	
1	44790UQ151P5198912070518	UQS117	NaN	NaN	UQS117	44790URZ201902250028	44790NTC201902250005	중로1류	42515.0	3900.0	44790	중로	1	2	PMI0002	
2	44790UQ151P5201808280065	UQS119	NaN	NaN	UQS119	44790URZ201911270024	44790NTC201911270017	중로3류	1041.0	211.0	44790	중로	3	2	PMI0004	
3	44790UQ151P5201808280076	UQS122	NaN	NaN	UQS122	44790URZ201911150023	44790NTC201911150016	소로3류	1185.0	411.0	44790	소로	3	10	PMI0004	
4	44790UQ151P5199903270383	UQS121	NaN	NaN	UQS121	44790URZ201911140020	44790NTC201911140013	소로2류	1012.0	290.0	44790	소로	2	44	PMI0004	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
260544	41110UQ151P5202011304086	UQS122	NaN	NaN	UQS122	41110URZ201601080283	41110NTC201601080343	소로3류/(소로3-732	226.0	103.0	41110	소로	3	732	PMI0004	
260545	41110UQ151P5202011304085	UQS120	NaN	NaN	UQS120	41110URZ201601110021	41110NTC201601110083	소로1류/(소로1-409	479.0	109.0	41110	소로	1	409	PMI0004	
260546	41110UQ151P5202011304084	UQS115	NaN	NaN	UQS115	41110URZ202007010341	41110NTC202007011512	대로2류/(대로2-17)	18576.0	1134.0	41110	대로	2	17	PMI0002	
260547	41110UQ151P5202011304082	UQS116	NaN	NaN	UQS116	41110URZ201603030318	41110NTC201603030362	대로3류/(대로3-68)	25470.0	1784.0	41110	대로	3	68	PMI0002	
260548	41110UQ151P5202011304000	UQS115	NaN	NaN	UQS115	41110URZ201603030308	41110NTC201603030362	대로2류/(대로2-15)	8898.0	636.0	41110	대로	2	15	PMI0002	

11 rows 260549 rows × 19 columns Static Output

NNC_SN	NTFC_SN	DGM_NM	DGM_AR	DGM_LT	SIGNGU_SE	GRAD_SE	ROAD_TY	ROAD_NO	ROAD_ROLE	DRAWING_NO	EXCUT_SE	CREATE_DAT	geometry
790URZ202002170001	44790NTC202002170001	소로1류	2866.0	516.0	44790	소로	1	1	PMI0004	NaN	EMA0002	NaN	POLYGON ((126.85109396989182 36.43998425368667...
790URZ201902250028	44790NTC201902250005	중로1류	42515.0	3900.0	44790	중로	1	2	PMI0002	NaN	EMA0001	NaN	POLYGON ((126.94764338587943 36.41572082414885...
790URZ201911270024	44790NTC201911270017	중로3류	1041.0	211.0	44790	중로	3	2	PMI0004	NaN	EMA0002	NaN	POLYGON ((126.95452509995475 36.41326972159591...
790URZ201911150023	44790NTC201911150016	소로3류	1185.0	411.0	44790	소로	3	10	PMI0004	NaN	EMA0003	NaN	POLYGON ((126.94748413409984 36.40889130064808,...
790URZ201911140020	44790NTC201911140013	소로2류	1012.0	290.0	44790	소로	2	44	PMI0004	NaN	EMA0003	NaN	POLYGON ((126.80156928996409 36.44532360272043,...
.	...	...	...	...	...	...	...	...	...	...	...	...	...
110URZ201601080283	41110NTC201601080343	소로3류/(소로3-732	226.0	103.0	41110	소로	3	732	PMI0004	NaN	EMA0001	NaN	POLYGON ((126.99544933861288 37.26954424697348...
110URZ201601110021	41110NTC201601110083	소로1류/(소로1-409	479.0	109.0	41110	소로	1	409	PMI0004	NaN	EMA0001	NaN	POLYGON ((126.99455510361983 37.26931254194859...
110URZ202007010341	41110NTC202007011512	대로2류/(대로2-17)	18576.0	1134.0	41110	대로	2	17	PMI0002	NaN	EMA0001	NaN	POLYGON ((127.01783897595706 37.27042415374969...
110URZ201603030318	41110NTC201603030362	대로3류/(대로3-68)	25470.0	1784.0	41110	대로	3	68	PMI0002	NaN	EMA0001	NaN	POLYGON ((127.05501400330743 37.24213399199607...
110URZ201603030308	41110NTC201603030362	대로2류/(대로2-15)	8898.0	636.0	41110	대로	2	15	PMI0002	NaN	EMA0001	NaN	POLYGON ((127.0686077399912 37.24162216489862,...

Table 7.18: 도로(현황) 데이터

7.2.3 기상환경변수

기상환경변수는 모두 기상청의 방재(AWS)데이터를 사용하였다 **??**. 방재 데이터에는 온도, 강수량, 풍압, 풍향, 기압 등 기상 정보가 포함되어 있다. 기상 데이터는 종관(ASOS)관측과 방제(AWS)를 고려하였다. 그러나 종관 관측은 관측지점이 105개밖에 없기에 과측 지점이 더 많아 데이터의 과적합을 예방할 수 있는 방제 데이터로 선택하였다. 황사(PM10)데이터도 고려해봤지만 교통류에 주는 영향이 미미할 것으로 판단하여 원시데이터에서 제외하였다. 기상청의 지점 정보 수집을 위해 날씨 데이터 지점 데이터 또한 추가하였다.

11 rows 369315 rows × 10 columns Static Output

	지점	지점명	일시	기온(°C)	풍향(deg)	풍속(m/s)	강수량(mm)	현지기압(hPa)	해면기압(hPa)	습도(%)
0	12	인연도(감)	2023-04-01 00:00	10.3	50.2	2.5	0.0	1011.3	1018.6	41.0
1	12	인연도(감)	2023-04-01 01:00	10.2	53.5	1.3	0.0	1011.3	1018.6	44.0
2	12	인연도(감)	2023-04-01 02:00	10.4	103.2	2.4	0.0	1011.3	1018.6	50.0
3	12	인연도(감)	2023-04-01 03:00	11.5	61.6	8.8	0.0	1010.9	1018.2	41.0
4	12	인연도(감)	2023-04-01 04:00	10.3	335.4	2.2	0.0	1010.4	1017.7	47.0
...	...	...	...	...	...	...	...	...	...	...
369310	994	심포	2023-04-29 20:00	11.4	320.6	10.8	0.0	1010.1	1010.5	82.0
369311	994	심포	2023-04-29 21:00	11.0	313.0	9.2	0.0	1010.9	1011.3	73.0
369312	994	심포	2023-04-29 22:00	10.9	332.9	7.6	0.0	1011.6	1012.0	75.0
369313	994	심포	2023-04-29 23:00	10.8	328.4	6.5	0.0	1011.8	1012.2	76.0
369314	994	심포	2023-04-30 00:00	10.6	331.5	8.4	0.0	1012.2	1012.6	76.0

Table 7.19: 방재 데이터

7.2.4 교통수요공공변수

행정안전부 지역별(행정동) 성별 연령별 주민등록 인구수를 수집하였다 **??**. 행정안전부 지역별(행정동) 성별 연령별 주민등록 인구수 데이터는 성별, 나이별 인구 분포를 시군구 단위로 측정하여 요구 유의 변수에 부합한다. 요구 유의 변수 외에 추가로 사고다발지점 데이터, 전국무인교통단속카메라표준데이터를 수집하였다 **??**, **??**. 두 데이터 모두 운전자에게 심리적 요인으로 작용할 수 있다고 판단하였다.

1-100 3614 rows × 230 columns

	행정기관코드	기준연월	시도명	시군구명	읍면동명	계	남자	여자	0세남자	1세남자	2세남자	3세남자	4세남자	5세남자	6세남자	7세남자	8세남자	9세남자	10세남
0	1111051500	2024-07-31	서울특별시	종로구	청운효자동	11178	5088	6090	23	19	19	24	32	26	23	48	42	38	
1	1111053000	2024-07-31	서울특별시	종로구	사직동	8994	3947	5047	8	15	22	15	16	20	26	25	35	48	
2	1111054000	2024-07-31	서울특별시	종로구	삼청동	2236	1069	1167	1	3	6	4	1	2	5	4	5	6	
3	1111055000	2024-07-31	서울특별시	종로구	부암동	9861	4263	4798	13	10	14	15	15	14	16	27	36	31	
4	1111056000	2024-07-31	서울특별시	종로구	평창동	17272	8009	9263	26	37	41	41	50	52	59	54	77	75	
5	1111057000	2024-07-31	서울특별시	종로구	무악동	7896	3626	4270	12	18	16	26	15	19	24	37	31	30	
6	1111058000	2024-07-31	서울특별시	종로구	교남동	9652	4421	5231	31	22	34	36	35	35	44	33	47	42	
7	1111060000	2024-07-31	서울특별시	종로구	가회동	3794	1781	2013	5	6	8	9	8	10	7	13	13	12	
8	1111061500	2024-07-31	서울특별시	종로구	종로1.2.3.4가동	6786	4046	2660	8	7	12	4	10	6	5	7	18	3	
9	1111063000	2024-07-31	서울특별시	종로구	종로5.6가동	5310	2798	2520	6	1	5	4	9	2	11	13	10	6	
10	1111064000	2024-07-31	서울특별시	종로구	이화동	6989	3263	3726	5	9	5	5	11	7	6	5	18	11	
11	1111065000	2024-07-31	서울특별시	종로구	혜화동	15931	7663	8268	17	30	23	9	24	23	21	22	35	35	
12	1111067000	2024-07-31	서울특별시	종로구	창신제1동	4478	2363	2115	3	7	10	5	8	2	2	4	5	7	
13	1111068000	2024-07-31	서울특별시	종로구	창신제2동	7424	3835	3589	5	3	8	6	10	7	13	9	5	15	
14	1111069000	2024-07-31	서울특별시	종로구	창신제3동	6315	3104	3211	13	12	16	13	16	15	18	21	18	13	
15	1111070000	2024-07-31	서울특별시	종로구	숭인제1동	5761	2786	2975	11	7	6	8	11	9	3	12	11	12	
16	1111071000	2024-07-31	서울특별시	종로구	숭인제2동	10096	4902	5194	11	7	14	11	9	7	7	12	10	14	
17	1114052000	2024-07-31	서울특별시	중구	소공동	2195	1023	1172	6	5	3	12	4	7	9	8	9	7	
18	1114054000	2024-07-31	서울특별시	중구	회현동	4356	2281	2075	6	7	9	6	10	7	7	10	8	12	
19	1114055000	2024-07-31	서울특별시	중구	명동	2578	1068	1510	2	4	3	4	1	0	6	0	6	3	
20	1114057000	2024-07-31	서울특별시	중구	필동	4315	2218	2097	6	9	2	8	4	3	7	4	11	13	
21	1114058000	2024-07-31	서울특별시	중구	장충동	4555	2372	2183	11	4	7	10	6	8	8	4	12	6	
22	1114059000	2024-07-31	서울특별시	중구	광희동	5780	2841	2939	9	5	6	7	6	3	1	10	5	9	
23	1114060500	2024-07-31	서울특별시	중구	을지로동	3129	1699	1430	15	13	9	5	5	3	2	6	4	2	
24	1114061500	2024-07-31	서울특별시	중구	신당동	7402	3611	3791	7	7	13	7	14	14	7	10	11	10	
25	1114062500	2024-07-31	서울특별시	중구	다산동	13117	6352	6765	16	17	16	22	21	19	16	23	28	33	
26	1114063500	2024-07-31	서울특별시	중구	약수동	16080	7411	8669	40	33	47	28	37	37	38	41	39	44	
27	1114064500	2024-07-31	서울특별시	중구	청구동	12835	4033	4802	39	31	26	27	30	22	27	34	37	35	
28	1114065000	2024-07-31	서울특별시	중구	신당제5동	10151	4925	5226	22	25	16	34	21	13	13	37	15	29	
29	1114066500	2024-07-31	서울특별시	중구	동화동	9887	4737	5150	22	32	15	30	31	25	29	35	43	30	
30	1114067000	2024-07-31	서울특별시	중구	황학동	12906	6062	6844	38	33	30	30	35	42	30	30	28	27	
31	1114068000	2024-07-31	서울특별시	중구	중림동	12121	5913	6208	32	31	48	52	42	55	53	48	52	43	
32	1117051000	2024-07-31	서울특별시	용산구	후암동	15932	7652	8280	17	30	31	24	37	36	28	36	59	44	
33	1117052000	2024-07-31	서울특별시	용산구	용산2가동	8799	4323	4476	11	18	5	8	9	12	10	16	23	11	
34	1117053000	2024-07-31	서울특별시	용산구	남영동	6792	4147	2645	8	3	5	3	7	0	8	10	10	9	

Table 7.20: 인구 데이터

1-100 13363 rows × 18 columns

	사고지역관리번호	사고연도	사고유형구분	위치코드	사고다발지역시도시군구	사고지역위치명	사고건수	사상자수	사망자수
Missing	0	0	0	0	0	0	0	0	0
Count	13363	13363	13363	13363	13363	13363	13363	13363	13363
	2013060 … 2022984	2012 … 2022	보행노인 37.3% 자전거 32.6% Other 30.1%	11041109 … 99999982	대구광역시 달서구1 0.3% 서울특별시 구로구1 0.3% Other 99.5%	10810 Unique values	1 … 27	1 … 32	0
0	2013098	2012	보행노인	11470001	서울특별시 양천구1	서울특별시 양천구 신월동(대흥빌딩	3	3	
1	2013098	2012	보행노인	11470002	서울특별시 양천구2	서울특별시 양천구 신정동(신정3...	3	4	
2	2013098	2012	보행노인	11500001	서울특별시 강서구1	서울특별시 강서구 화곡동(홍익병...	3	3	
3	2013098	2012	보행노인	11500002	서울특별시 강서구2	서울특별시 강서구 화곡동(강서구청	3	3	
4	2013098	2012	보행노인	11500003	서울특별시 강서구3	서울특별시 강서구 화곡동(화곡시거	4	4	
5	2013098	2012	보행노인	11500004	서울특별시 강서구4	서울특별시 강서구 염창동(태진빌딩	4	4	
6	2013098	2012	보행노인	11500005	서울특별시 강서구5	서울특별시 강서구 내발산동(우장산	5	6	
7	2013098	2012	보행노인	11380003	서울특별시 은평구3	서울특별시 은평구 불광동(불광동...	3	3	
8	2013098	2012	보행노인	11380004	서울특별시 은평구4	서울특별시 은평구 증산동(정솔식당	3	3	
9	2013098	2012	보행노인	11380005	서울특별시 은평구5	서울특별시 은평구 녹번동(선학수련	4	4	
10	2013098	2012	보행노인	11380006	서울특별시 은평구6	서울특별시 은평구 신사동(신사시거	4	5	
11	2013098	2012	보행노인	11380007	서울특별시 은평구7	서울특별시 은평구 대조동(역촌시거	4	4	
12	2013098	2012	보행노인	11380008	서울특별시 은평구8	서울특별시 은평구 불광동(지현빌딩	4	5	
13	2013098	2012	보행노인	11380009	서울특별시 은평구9	서울특별시 은평구 불광동(연희초...	4	4	
14	2013098	2012	보행노인	11380010	서울특별시 은평구10	서울특별시 은평구 대조동(대조...	5	8	
15	2013098	2012	보행노인	11380011	서울특별시 은평구11	서울특별시 은평구 불광동(불광전화	5	7	
16	2013098	2012	보행노인	11380012	서울특별시 은평구12	서울특별시 은평구 대조동(신화은...	7	9	
17	2013098	2012	보행노인	11410001	서울특별시 서대문구1	서울특별시 서대문구 홍제동(홍안빌	3	3	
18	2013098	2012	보행노인	11410002	서울특별시 서대문구2	서울특별시 서대문구 남가좌동(홍...	3	3	
19	2013098	2012	보행노인	11410003	서울특별시 서대문구3	서울특별시 서대문구 홍은동(홍은...	5	5	
20	2013098	2012	보행노인	11410004	서울특별시 서대문구4	서울특별시 서대문구 창천동(신촌...	5	5	
21	2013098	2012	보행노인	11440001	서울특별시 마포구1	서울특별시 마포구 아현동(골라빌...	3	3	
22	2013098	2012	보행노인	11440002	서울특별시 마포구2	서울특별시 마포구 신수동(우마키서	4	4	
23	2013098	2012	보행노인	11305003	서울특별시 강북구3	서울특별시 강북구 미아동(양천시장	7	7	
24	2013098	2012	보행노인	11320001	서울특별시 도봉구1	서울특별시 도봉구 방학동(LG대경...	3	3	
25	2013098	2012	보행노인	11320002	서울특별시 도봉구2	서울특별시 도봉구 창동(창동역사촌	3	3	
26	2013098	2012	보행노인	11320003	서울특별시 도봉구3	서울특별시 도봉구 쌍문동(숭이초교	3	3	
27	2013098	2012	보행노인	11320004	서울특별시 도봉구4	서울특별시 도봉구 도봉동(은누리...	3	3	
28	2013098	2012	보행노인	11320005	서울특별시 도봉구5	서울특별시 도봉구 창동(창원초교	3	3	
29	2013098	2012	보행노인	11320006	서울특별시 도봉구6	서울특별시 도봉구 방학동(아람미...	4	4	

Table 7.21: 사고다발지점 데이터

1-100 24089 rows × 18 columns

	무인교통단속카메라관리번호	시도명	시군구명	도로종류	도로노선명	도로노선방향	소재지지번주소	위도	경도
Missing Count	0	0	0	0	0	0	0	0	0
Count	24089	24089	24089	24089	24089	24089	24089	24089	24089
Unique values	22713	경기도 19.8% 충청남도 9.4% Other 70.9%	서구 2.7% 북구 2.5% Other 94.9%	기타 59.5% 일반국도 17% Other 23.4%	중앙로 0.8% 동해대로 0.5% Other 98.8%	1 35.8% 3 31.2% Other 33%	20410	21164	21163

0	고현-0021	경상남도	거제시	기타	중곡로	03	경상남도 거제시 고현동 988-32	34.8959	128.62993
1	81	경기도	성남시	기타	미금일로79번길	03	경기도 성남시 분당구 구미동 96-:	37.349860	127.112129
2	80	경기도	성남시	기타	미금일로79번길	03	경기도 성남시 분당구 구미동 65	37.347268	127.113128
3	계양지구2	경상북도	경산시	시도	장산로	03	경상북도 경산시 계양동 705	35.81772812	128.7521716
4	역전네거리	경상북도	경산시	시도	경산로	03	경상북도 경산시 사동동 78-11	35.81981828	128.728834
5	대학로4	경상북도	경산시	지방도	대학로	03	경상북도 경산시 조영동 314-1	35.83781346	128.7553064
6	1	강원특별자치도	태백시	일반국도	황지로	3	강원특별자치도 태백시 황지동 234	37.17777711	128.9864657
7	2	강원특별자치도	태백시	일반국도	연지로	3	강원특별자치도 태백시 황지동 265	37.17527562	128.9892374
8	3	강원특별자치도	태백시	일반국도	시장북길	3	강원특별자치도 태백시 황지동 38-	37.17308715	128.992276
9	4	강원특별자치도	태백시	일반국도	태백로	3	강원특별자치도 태백시 황지동 36-	37.17395519	128.9935597
10	5	강원특별자치도	태백시	일반국도	반양로	3	강원특별자치도 태백시 황지동 31-	37.17181828	128.9924663
11	6	강원특별자치도	태백시	일반국도	문화로	3	강원특별자치도 태백시 황지동 81-	37.16646655	128.9915421
12	7	강원특별자치도	태백시	일반국도	광장로	3	강원특별자치도 태백시 황지동 270	37.17684172	128.9845886
13	N158	서울특별시	금천구	특별시도	벚꽃로18길	3	서울특별시 금천구 독산동 999-11	37.46822619	126.8908158
14	N159	서울특별시	금천구	특별시도	시흥대로	3	서울특별시 금천구 시흥동 1026	37.44614238	126.9010582
15	N160	서울특별시	금천구	특별시도	독산로54길	3	서울특별시 금천구 독산동 992-8	37.473621	126.905796
16	N161	서울특별시	금천구	특별시도	독산로77	3	서울특별시 금천구 시흥동 841-16	37.454324	126.905803
17	N162	서울특별시	금천구	특별시도	독산로23길	3	서울특별시 금천구 시흥동 841-19	37.454821	126.904865
18	N165	서울특별시	금천구	특별시도	독산로36길	3	서울특별시 금천구 시흥동 883-19	37.459199	126.905609
19	N166	서울특별시	금천구	특별시도	시흥대로48길	3	서울특별시 금천구 시흥동 936-8	37.447774	126.903984
20	H6678	경기도	광주시	지방도	이배재로	01	경기도 광주시 탄벌동 182-4	37.41930235	127.2472513
21	J1878	경기도	군포시	기타	송부로	3	경기도 군포시 부곡동 871-6	37.32045177	126.9291189
22	J1879	경기도	군포시	기타	송부로	3	경기도 군포시 부곡동 870-1	37.32121566	126.9290776
23	J1880	경기도	군포시	일반국도	군포로	3	경기도 군포시 부곡동 768-5	37.33282263	126.9269545
24	J1883	경기도	군포시	기타	호수로	3	경기도 군포시 대야미동 229-3	37.32809471	126.9164667
25	J1884	경기도	군포시	기타	호수로	3	경기도 군포시 대야미동 234-6	37.3283986	126.9164685
26	J1887	경기도	군포시	기타	이당로	3	경기도 군포시 당정동 573-3	37.3480307	126.958293
27	J1892	경기도	군포시	기타	산본로	3	경기도 군포시 당정동 126-11	37.35464024	126.9574279
28	J1897	경기도	군포시	기타	고산로	3	경기도 군포시 산본동 1222	37.36608974	126.9285583
29	J1898	경기도	군포시	기타	고산로	3	경기도 군포시 산본동 1063-2	37.3682715	126.9321753
30	J1899	경기도	군포시	기타	고산로	3	경기도 군포시 산본동 1263	37.37064584	126.935501

Table 7.22: 전국무인교통단속카메라표준데이터

7.2.5 차량현황변수

국토교통부 자동차 등록 현황을 사용하였다 **??**. 국토교통부 자동차 등록 현황 데이터는 차량의 종류별 통계와 전체 통계를 모두 포함한다. 경찰청 운전면허소지자 연령별 현황 데이터도 사용하였다 **??**.

1	월(Monthly)	시도명	시군구	승용	승용	승용	승용	승합	승합	승합	승합	화물	화물	화물	화물	특수	특수	특수	특수	총계	총계	총계
2	월(Monthly)	시도명	시군구	관용	자가용	영업용	계	관용	자가용	영업용	계	관용	자가용	영업용	계	관용	자가용	영업용	계	관용	자가용	영업용
3	2024-08	서울	강남구	160	218	724	16	964	235	848	61	3	923	570	4	554	189	13	621	2	423	16
4	2024-08	서울	강동구	110	132	052	4	939	137	101	41	3	258	679	3	978	145	11	902	2	544	14
5	2024-08	서울	강북구	107	60	092	2	888	62	287	69	2	092	714	2	875	130	6	834	1	288	8
6	2024-08	서울	강서구	136	167	155	9	214	176	505	45	4	178	1	388	5	531	167	16	212	4	265
7	2024-08	서울	관악구	89	99	193	3	024	102	306	45	2	978	345	3	368	139	9	847	2	232	11
8	2024-08	서울	광진구	91	80	461	3	105	83	657	45	2	085	326	2	456	172	8	570	1	599	10
9	2024-08	서울	구로구	108	115	461	9	532	125	101	62	3	941	578	4	581	128	13	832	2	637	16
10	2024-08	서울	금천구	60	71	811	3	385	75	256	32	2	619	451	3	182	74	10	747	2	299	13
11	2024-08	서울	노원구	84	129	663	5	611	135	358	54	3	289	770	4	113	175	9	783	2	161	12
12	2024-08	서울	도봉구	92	76	344	4	639	81	075	48	2	252	578	2	878	129	7	926	2	850	10
13	2024-08	서울	동대문구	106	84	542	2	912	87	568	55	2	541	131	2	727	131	9	722	2	196	12
14	2024-08	서울	동작구	115	94	686	1	119	95	928	45	2	189	130	2	364	135	6	452	1	196	7
15	2024-08	서울	마포구	95	101	855	4	234	106	184	55	2	493	639	3	187	177	9	240	2	174	11
16	2024-08	서울	서대문구	209	77	491	3	436	81	336	106	1	983	482	2	491	134	5	821	1	805	7
17	2024-08	서울	서초구	176	153	184	3	885	157	165	134	2	948	646	3	720	183	10	618	4	727	15
18	2024-08	서울	성동구	132	82	131	9	166	91	429	46	1	792	982	2	820	179	8	169	1	872	10
19	2024-08	서울	성북구	96	105	369	2	805	107	478	50	2	781	1	103	3	934	138	8	507	1	714
20	2024-08	서울	송파구	130	206	948	7	592	214	678	101	3	962	1	635	5	698	160	17	466	4	777
21	2024-08	서울	양천구	131	127	351	4	393	131	875	61	3	036	730	3	827	147	11	812	2	928	14
22	2024-08	서울	영등포구	128	118	418	2	918	121	464	61	3	621	180	3	862	168	13	288	4	470	17
23	2024-08	서울	용산구	260	63	683	1	279	65	222	118	1	767	77	1	962	154	4	983	962	6	099
24	2024-08	서울	은평구	101	114	108	4	038	118	247	47	2	931	743	3	721	143	9	809	2	863	12
25	2024-08	서울	종로구	1	654	39	397	236	41	287	1	540	1	432	249	3	221	340	3	645	1	564
26	2024-08	서울	중구	301	41	335	1	685	43	241	857	1	747	231	2	835	177	5	226	1	194	6
27	2024-08	서울	중랑구	78	91	894	5	967	97	939	33	2	968	580	3	581	135	11	565	2	792	14
28	2024-08	서울	계	4	749	2	653	548	117	206	2	775	503	3	811	68	798	14	777	87	386	3
29	2024-08	부산	강서구	117	68	089	2	114	70	328	86	1	646	310	2	042	130	16	244	2	752	19
30	2024-08	부산	금정구	99	65	067	2	448	67	614	62	2	008	870	2	940	103	9	437	3	101	12
31	2024-08	부산	기장군	80	74	323	832	75	235	49	2	008	549	2	686	120	10	767	1	284	12	171
32	2024-08	부산	남구	108	81	357	3	066	84	531	58	2	062	497	2	617	58	8	801	4	353	13
33	2024-08	부산	동구	82	44	488	1	708	46	278	47	844	172	1	063	68	3	593	2	387	6	048
34	2024-08	부산	동래구	71	85	429	1	642	87	142	27	1	865	371	2	263	65	9	887	1	509	11

Table 7.23: 국토교통부 자동차 등록 현황 데이터

	구분	1종대형	1종보통	1종소형	대형견인	소형견인	구난	2종보통	2종소형	원자
1	16	0	0	0	0	0	0	0	0	2197
2	17	0	0	0	0	0	0	0	0	4854
3	18	0	28078	0	0	0	0	22321	131	2728
4	19	729	112365	0	0	0	2	189397	180	2259
5	20	3896	143684	0	5	1	18	156613	243	2794
6	21	7723	148759	0	7	4	14	175292	264	2599
7	22	12819	165957	0	7	5	21	208607	214	2138
8	23	16137	192644	0	13	5	37	241373	301	2192
9	24	18264	208872	0	21	16	36	263078	344	2296
10	25	21487	230108	0	19	11	35	283466	338	2383
11	26	23970	249077	0	31	12	29	291867	310	2039
12	27	25989	267579	0	28	19	23	291873	293	1670
13	28	28045	291295	0	17	13	24	293997	292	1523
14	29	28548	291228	0	18	24	26	288561	249	1386
15	30	26588	280108	0	31	34	29	262657	202	1245
16	31	26484	283771	0	31	26	28	260536	202	1246
17	32	26125	295667	0	24	36	19	252035	184	1266
18	33	26398	308034	0	27	43	20	245632	158	1273
19	34	28892	317697	0	18	46	16	238424	172	1377
20	35	31310	341256	0	19	54	20	233959	173	1369
21	36	32485	358596	0	28	67	27	225802	161	1451
22	37	40381	417799	0	34	68	23	238711	183	1462
23	38	46465	468213	0	36	71	19	249927	187	1644
24	39	49557	484485	0	35	62	21	245610	182	1511
25	40	51389	484848	0	39	67	21	240257	185	1444
26	41	53827	477264	0	56	79	18	238455	198	1418
27	42	49771	415602	0	42	63	21	204844	156	1362
28	43	53804	448138	0	52	68	27	225038	165	1335
29	44	54425	427848	0	76	88	17	220435	129	1321
30	45	57749	435282	0	58	43	19	230116	147	1341
31	46	65222	469061	0	98	53	24	247043	185	1486
32	47	68593	479791	0	101	58	17	259186	147	1523
33	48	69742	479008	0	101	47	17	257754	151	1677
34	49	75260	499534	0	93	58	19	269348	155	1778
35	50	73868	482781	0	94	46	24	252318	191	1954

Table 7.24: 경찰청 운전면허소지자 연령별 현황 데이터

7.2.6 원시 데이터 요약

원시 데이터의 요약은 다음과 같다 **??**.

7.2.7 데이터 특이사항

도로(현황)데이터는 shp파일이었기에 geopandas의 기능을 이용해 데이터 출처인 디지털 트윈 국토에 지정되 있던 epsg:4326 좌표계로 csv파일 인코딩 하였다. (데이터 제공자인 브이월드에서는 5186 좌표계를 지정하였다. 그러나 위, 경도 데이터로 변환하기 위해 epsg:4326 좌표계를 사용하였다.)

한국도로공사의 혼잡빈도 형식은 CSV 데이터이지만 파일에 확장자가 붙어 있지 않아 직접 .csv확장자를 붙혔다[21].

ITS 국가교통정보센터의 표준노드링크 데이터 또한 shp파일을 geopandas를 이

[21]참고: 붙혔을 때 형식의 변화는 일어나지 않았다, 아마 다운로드 과정에서의 확장자 손실이 있었던 것으로 추정된다.

항목	수집기간	수집주기	용량
ITS 교통 소통 정보	2023.1 – 2023.10	5분	약 500GB
ITS 교통 돌발 정보	2023.1 – 2023.10	분	약 100MB
기상청 방재(AWS) 데이터	2023.1 – 2023.10	시간	약 200MB
ITS 표준노드링크 데이터	2023.8	년	약 1.4 GB
한국도로공사 혼잡빈도	2023.1 – 2023.10	5분	약 4GB
기상청 관측 지점 데이터	2023.8	년	약 16KB
전국교통사고다발지역표준데이터	2023.8	년	약 15MB
전국무인교통단속카메라표준데이터	2023.8	년	약 14MB
행정안전부 지역별 주민등록 인구수	2023.10	년	약 3MB
도로(현황) 데이터	2023.10	년	약 3GB
국토교통부 자동차 등록 현황	2024.06	년	약 40KB
경찰청 운전면허소지자 연령별 현황 데이터	2020.12	년	약 3.3KB

Table 7.25: 원시데이터 요약

용하여 epsg:4326 좌표계의 csv파일로 변환하였다.

ITS 국가교통정보센터의 교통소통정보 데이터 중 2023-04-24 날짜의 데이터는 파일데이터를 압축해제하는 단계에서 오류가 발생하여 분석 데이터 구성에서 제외하였다.

다음은 전체 특이사항 항목이다 **??**.

7.3 데이터 전처리

본 연구에서는 위에서 개발한 세가지 모델의 데이터 처리 양식에 맞게 일련의 데이터 전처리 과정을 수행하였다. 데이터 전처리 과정은 유의 변수 설정, 결측 값 보정, 데이터 병합의 과정으로 구성되어 있다.

항목	내용	처리
도로(현황)데이터	shp 형식	.csv 파일로 형식 변환
혼잡빈도	확장자 소실	.csv 확장자 첨부
표준노드링크	shp 형식	.csv 파일로 형식 변환
교통소통정보	2024.04.24 데이터 소실	데이터 구성에서 제외

Table 7.26: 데이터 특이사항

7.3.1 데이터 전처리 개요

분석 데이터 생성은 다음 관점에 따라 중점적으로 구성되었다.

첫 번째, ITS 국가교통정보센터의 표준노드링크 데이터를 중점적으로 구성되었다. ITS 국가교통정보센터의 표준노드링크 데이터는 도로 정보를 도로와 지점들로 나눠 더욱 지리적 정보를 효율적이고 직관적으로 담았다. 링크 데이터와 노드 데이터로 나눠져 있으며, 각 데이터는 링크 아이디와 노드 아이디로 연결되어 있다. 또 국가교통정센터에서 제공하는 타 중요 데이터인 소통 정보와 돌발 정보도 주요 지리 정보로 링크 아이디를 사용하기에 표준노드링크 데이터는 본 연구에서 매우 중요한 역할로 자리하고 있다.

두번째, 한국도로공사의 혼잡빈도 데이터를 중점적으로 구성되었다. 혼잡빈도 데이터는 교통량, 속도, 차로 번호, 혼잡빈도의 데이터들로 이루어져 있다. 데이터의 주요 변수가 모두 위에서 개발한 세가지 결합 모델에 필수적인 데이터이기에 지리 정보 만큼이나 주요한 데이터이다. 그러나 혼잡빈도 데이터는 지리 변수로 표준노드링크가 아닌 VDS코드를 사용한다. 이에 대한 대안으로 본 연구에서는 도로명과 주소를 통해 혼잡빈도 데이터를 병합하였다.

세번째, ITS 국가교통정보센터의 교통소통정보 데이터를 중점적으로 구성되었다. 교통소통정보 데이터는 본 연구에서 수집하고 사용한 모든 데이터 중 가장 용량이

큰 데이터이다. 대용량 데이터는 분석 데이터의 과적합을 막아주며, 데이터의 다양성을 통해 모델의 학습 성능을 제고할 수 있다. 혼잡빈도와 마찬가지로 속도 데이터를 포함하고, 혼잡빈도 데이터에 대해 데이터 증식을 위해 사용되었다.

7.3.2 유의 변수 설정

요구 유의 변수 설정은 위의 데이터 템플릿 절에서 완료된 바 있다. 그러나 본 연구의 실증 연구에서는 요구 유의 변수에 더해 몇가지의 추가 변수를 사용하였다. 이는 모델의 성능 제고를 위한 것이다.

국가교통정보센터의 교통 소통 정보 데이터에서는 변수 식별을 위한 날짜 데이터인 생성일과 생성시분, 지리적 변수 식별을 위한 링크 아이디, 실질 데이터인 통행속도를 사용하였다. 국가교통정보센터의 교통 돌발 정보 데이터에서는 날짜 변수인 돌발일시, 실질적 돌발 상황 변수인 돌발 내용, 지리 데이터는 소통정보와의 병합을 위한 변수인 링크 아이디를 사용하였다. 위, 경도 데이터는 돌발 데이터의 쓰임세인 보조 변수에 부합하지 않기에 배제하였다. 혼잡빈도 데이터는 날짜 데이터, 변수 집계년월, 집계시분과, 실질 변수 평균 교통량과 평균 속도, 그리고 혼잡빈도 수, 마지막으로 간접변수인 차로 번호를 사용하였다.

표준노드링크의 링크 데이터는 링크 아이디, 노드 데이터와의 연결을 위한 시작 노드 와 끝 노드 아이디를 중점으로 사용하였다. 그리고 차로 수, 제한 속도, 위치 데이터를 사용하였다. 노드 데이터는 노드 아이디와, 위치 데이터를 교차로 지점을 사용하였다. 도로(현황)데이터는 링크 데이터가 결측되었을 때, 보정 자료로 활용하기 위하여 도로 위치, 도로 타입을 사용하였다.

기상청의 방재 데이터는 예측 정확도를 위해 모든 기상 변수를 사용하였다. 기상청 지점 정보는 결합을 위한 위, 경도 데이터와 지점 코드 데이터 만을 사용하였다.

행정안전부 지역별(행정동) 성별 연령별 주민등록 인구수 데이터는 연산 속도를 위해 50대 까지의 데이터만을 사용하였고, 추가로 합계 데이터 까지를 포함하였다.

인구 수가 0이 나이 분포는 삭제하여 데이터를 축소시켰다.

국토교통부 자동차 등록 현황, 국토교통부 자동차 등록 현황 데이터의 차량의 종류별 통계, 경찰청 운전면허소지자 연령별 현황 데이터는 실질 데이터 만을 포함하였다. 따라서 모든 변수를 유의 변수에 포함하였다.

본 연구에서 추가로 사용한 공공 데이터인 전국사고다발지역 표준 데이터와 전국무인교통단속카메라표준데이터는 실질 데이터인 사고다발지역과 단속카메라의 위치 데이터만 사용하고, 종류와 같은 부가적 변수는 드랍하였다.

전체 유의 변수

전체 유의 변수는 다음과 같다 **??**.

변수 범위	유의 변수 목록
교통직접변수	속도, 교통량, 혼잡빈도수, 통행시간, 돌발일시, 돌발내용
도로환경변수	링크아이디, 노드아이디, 시작노드, 끝노드, 차로개수, 차로번호, 도로길이, 위, 경도, 교차로 개수
기상환경변수	기온, 풍향, 풍속, 강수량, 현지기압, 해면기압, 습도
교통수요변수	사고다발지점 유형, 지점 사고 건수, 지점 사상자 수, 교통단속카메라 단속 유형, 제한속도, 보호구역구분
차량현황변수	합계 인구, 남성 인구, 여성 인구, 나이별 인구

Table 7.27: 유의변수 목록

7.4 결측치 보정

7.4.1 결측치 탐색

교통 데이터는 환경적 요인과 인적 요인 등에 의해 간헐적으로 결측이 발생할 수 있다. 이러한 결측치는 모델에서 처리하지 못하기 때문에 데이터 보정이 필요하다. 결측치의 탐색은 파이썬의 결측치 시각화 패키지인 Missingo를 사용하였다. Python과 pandas를 이용하여 각 데이터의 결측치를 탐지 한 뒤 그 보고를 csv형식으로 저장하였다. 이 과정에서 소통 정보와 같이 매우 큰 용량의 데이터는 pandas와 함께 데이터 로드에 병렬 연산을 지원하는 dask를 사용하였다.

7.4.2 결측치 보정

ITS 국가교통정보센터의 교통소통데이터는 dask를 이용해서 데이터를 로드 한 뒤, 최적화를 위해 각 컬럼의 dtype을 바꾸고, 도로관리 기관을 드랍하였다. 그 외에는 모든 데이터에 결측치가 존재하지 않았다.

ITS 국가교통정보센터의 교통돌발데이터는 Missingno를 사용하여 시각적으로 결측치를 탐색하였다. 그래프의 시각적 분석 결과, 특별한 결측치 간 상관관계는 없었다 **??**. 그러나 결측치가 한 행에서 동시에 발생하는 경향을 띄었다 **??**. 이에 따라 결측치가 존재하는 행을 모두 제거하였다.

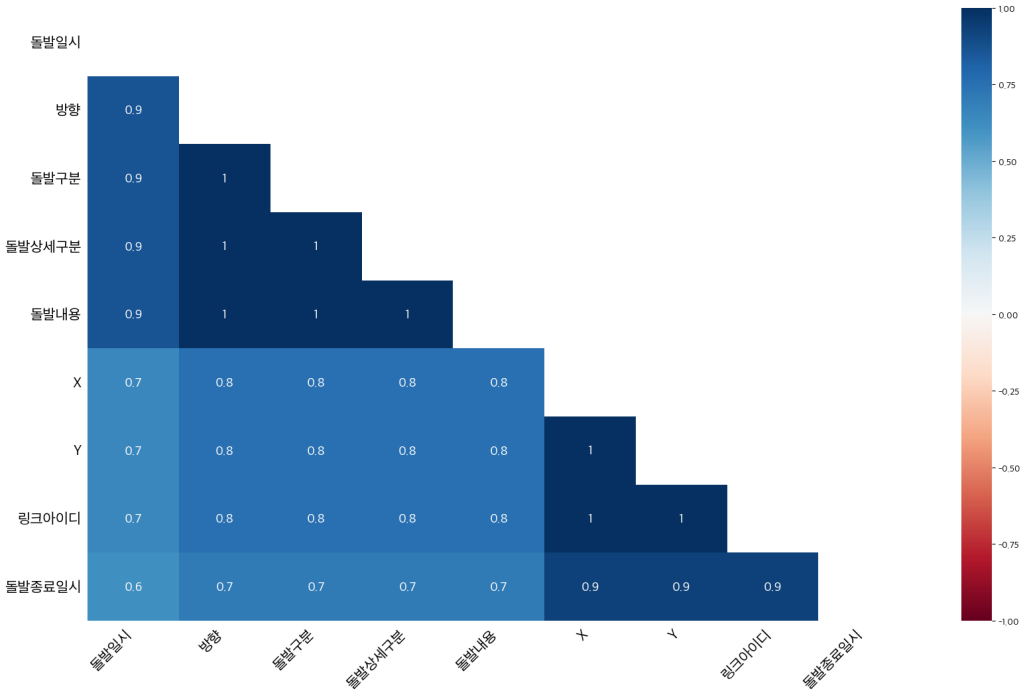

Figure 7.26: 돌발 데이터 Missingno heatmap graph

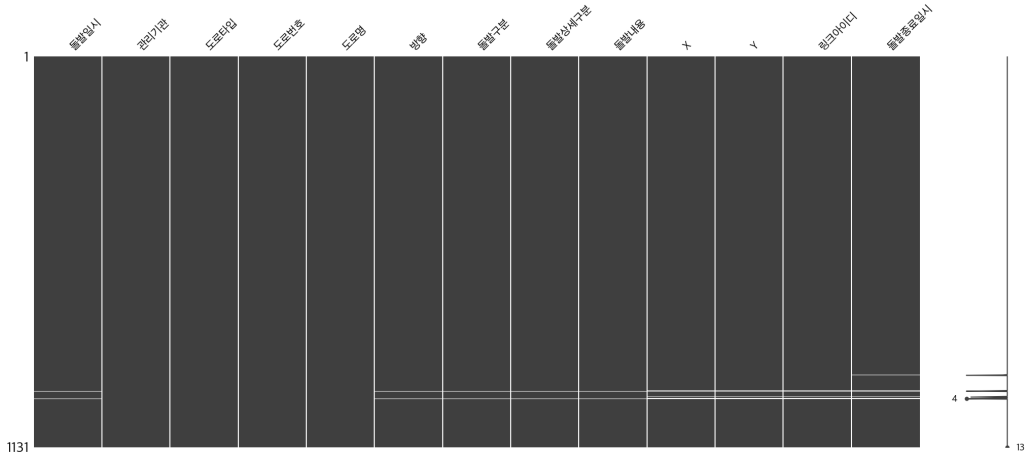

Figure 7.27: 돌발 데이터 Missingno matrix graph

방재 데이터 또한 Missingo를 통해 탐색하였다. 방제 데이터의 결측치를 missingno를 통해 분석한 결과, 풍향에 관련된 두 변수, 그리고 기온에 관련된 두 변수가 각각 1.0, 0.9의 확률로 동시에 결측치가 발생한다는 것을 heatmap과 matrix 그래프를 통해 조사하였다 **??**, **??**. 그리고, 방제 데이터는 날짜를 기준으로 하는 시계열 데이터라는 사실을 고려하여, Python의 데이터 과학 패키지인 Scikit-learn의 KNN Imputer을 통해 결측치를 보정하였다 [22]. KNN 알고리즘에서 n_ neighbor 값은 3으로 하였다.

[22]사이킷 런(Scikit-learn)의 K-최근접 이웃(K-nearest neighbors, KNN) 알고리즘을 활용한 결측치 처리법은 결측치를 단순히 제거하거나 대체하는 전통적 방식과 달리 분석을 통한 방식을 사용한다. KNN을 사용한 결측치 대체(imputation) 기법은 관측치 간의 유사성을 바탕으로 결측값을 추정하는 방식으로, 각 결측치를 해당 값이 속한 이웃 데이터의 정보로 보완한다. 구체적으로, KNN을 통한 결측치 처리는 먼저 결측값이 존재하는 샘플과 유사한 패턴을 보이는 다른 샘플을 찾는 과정을 거친다. 이 때, 결측치를 가진 샘플과 가장 가까운 K개의 이웃(n_ neighbor)을 선정한다. 그 후, 이웃들의 값을 기반으로 결측된 값을 대체하는데, 이 과정에서 평균값 또는 가중평균값을 사용하여 결측치를 채운다.

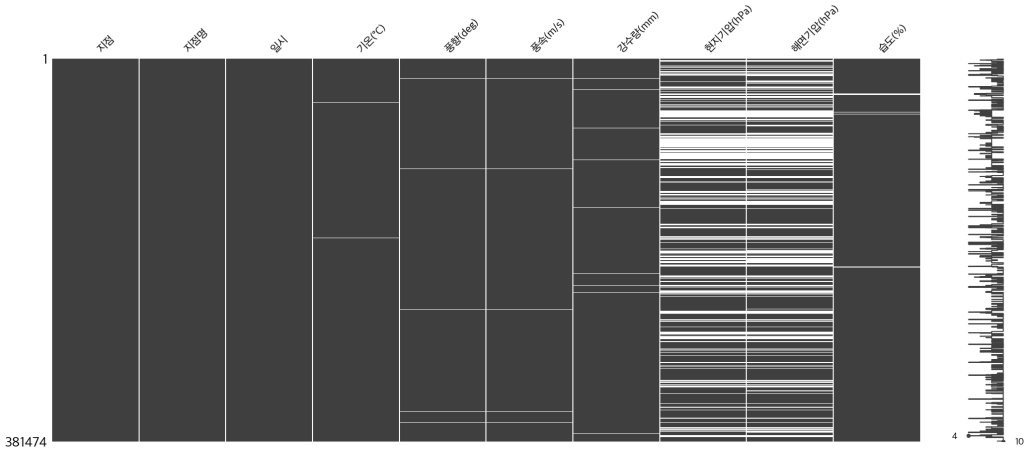

Figure 7.28: 방재(AWS)데이터 Missingno matrix graph

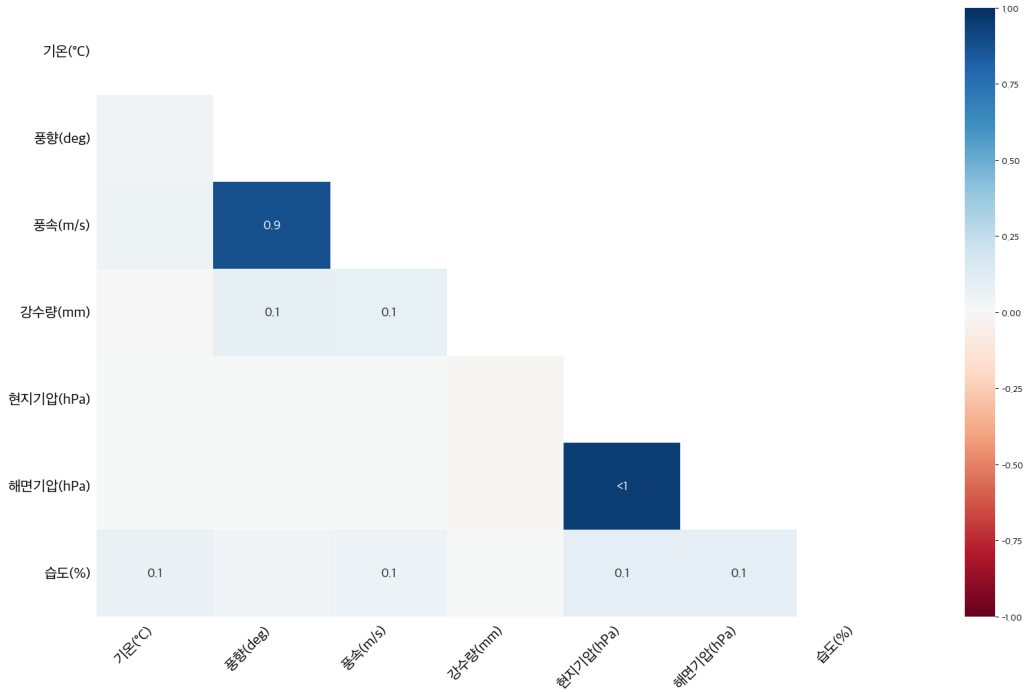

Figure 7.29: 방재(AWS)데이터 Missingno heatmap graph

도로현황 데이터는 MLSFC_ Cl등 몇몇 열의 결측치 양이 너무 많았다(70% 이상) **??**, **??**. 따라서 해당 열들을 드랍했다. 나머지 결측치들은 도로데이터였고, 연속하지 않은 데이터라는 사실을 고려하여 해당되는 행을 모두 삭제하여 결측치를 제거 하였다[23].

[23]도로 형태 등 중요 데이터가 많이 결측되어있었다. 따라서 결측치가 존재하는 행들은 이용가치가 없는 행들이었다.

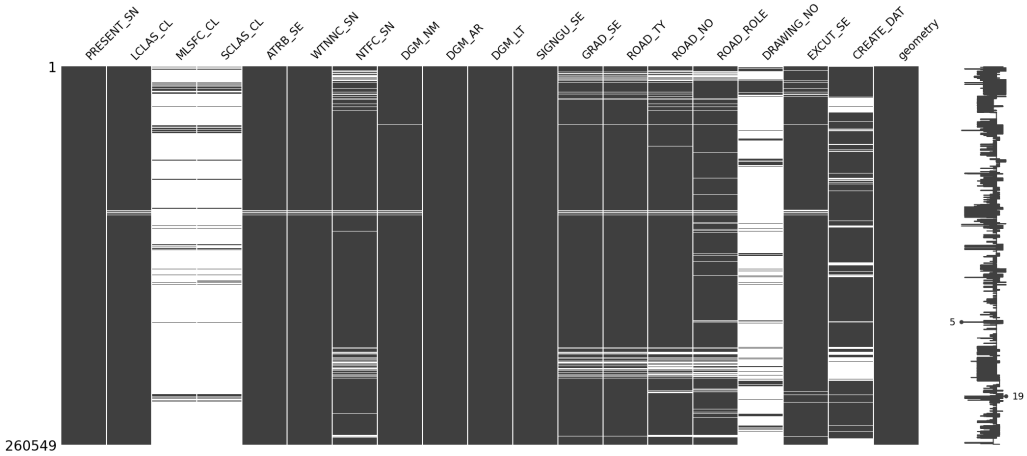

Figure 7.30: 도로(현황)데이터 Missingno matrix graph

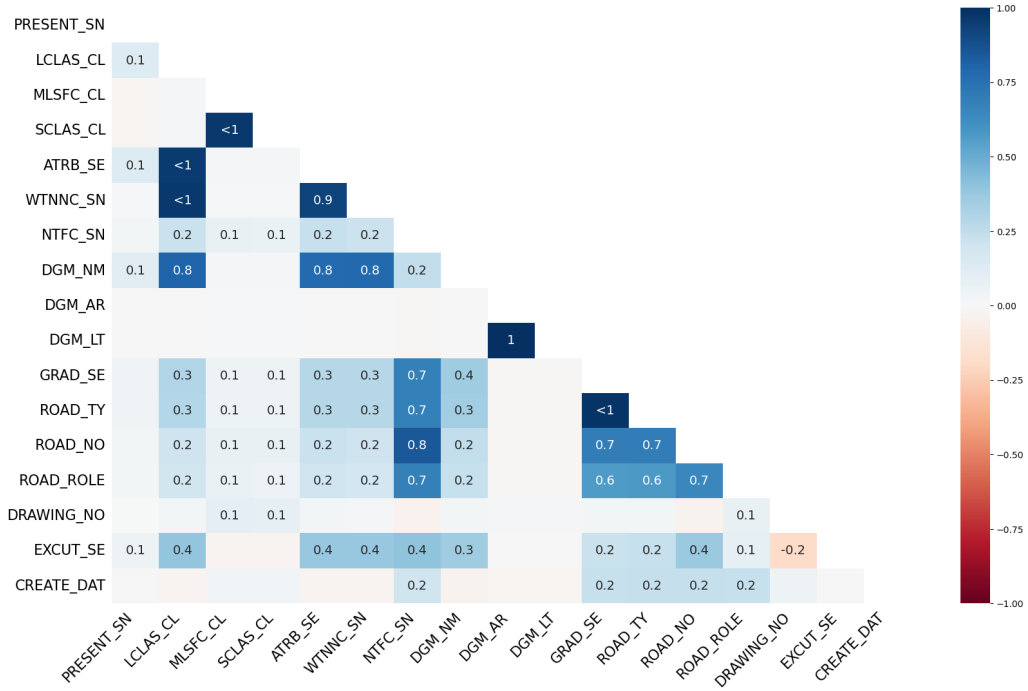

Figure 7.31: 도로(현황)데이터 Missingno heatmap graph

표준노드링크 데이터는 결측치 측정 중 dtype 인코딩이 dask 와 pandas에서 서로 같지 않아 연산이 진행되지 않는 문제가 발견되었다. 그러나 해당 열은 불필요하여 드랍 한뒤 진행하였다. 노드와 링크 데이터 모두 다른 곳은 결측치가 존재하지 않았지만 REMARK와 HITS_ TYPE, 그리고 HISTREMARK 열에서는 결측치가 90퍼 이상이기에 그 열들만 드랍하였다 **??**, **??**. 혼잡빈도 데이터는 Unnamed:13이라는 더미 데이터 열 말곤 결측치가 없어 해당 열만 드랍하였다.

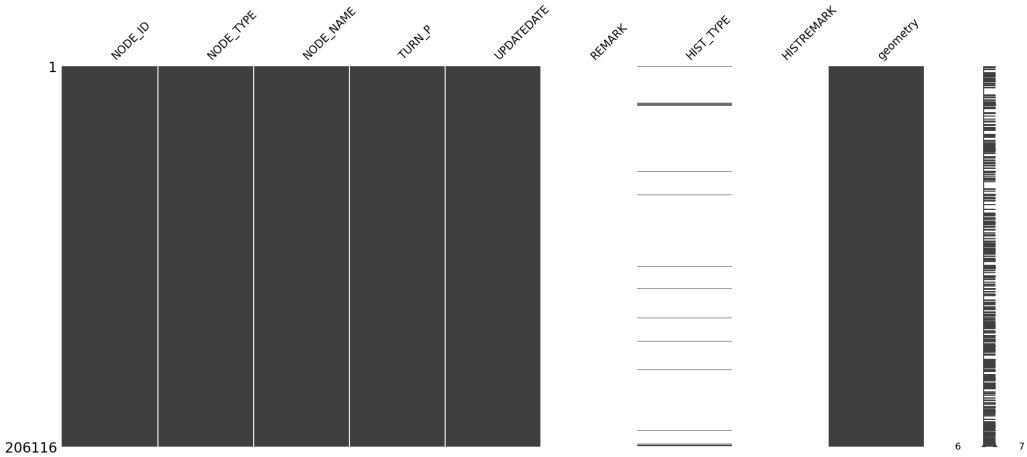

Figure 7.32: 노드 데이터 Missingno matrix graph

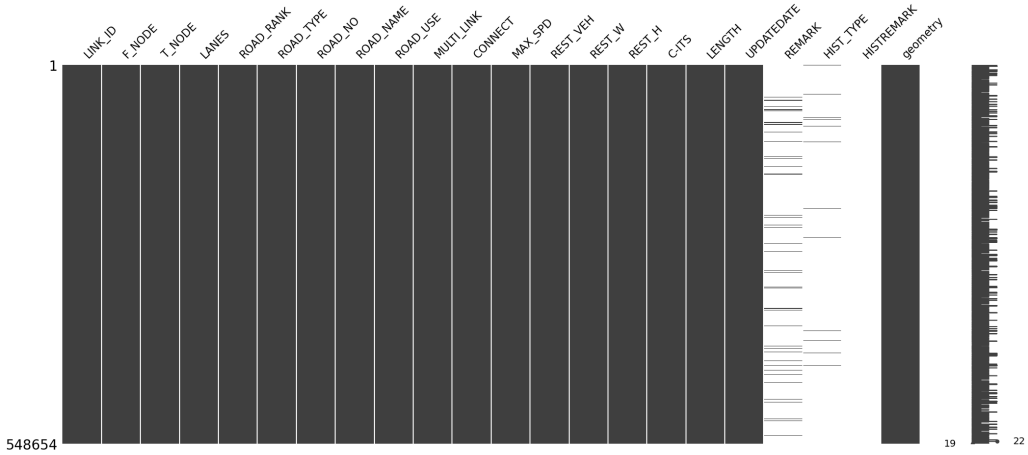

Figure 7.33: 링크 데이터 Missingno matrix graph

전국 무인교통단속 카메라 자료에서는 특정 열과 행에서 결측치가 한꺼번에 발생하는 양상을 보였다. 결측치가 과반수 이상 발견된 소재지 도로명과 설치연도을 드랍하였다. 나머지 결측치는 그 정도가 미미하여 행으로서 드랍하였다. 그 외 결측치는 발생하지 않았다.

7.5 데이터 통합

데이터 통합은 프라이머리 키의 상관관계를 고려하여 두개일 데이터 씩 차례대로 통합하였다. 통합은 dask와 pandas데이터를 사용하였으며, 이를 포함한 파이썬을 사용하여 데이터 통합 프로그램을 개발하였다. 이 프로그램은 이제부터 살펴볼 데이터 통합의 방법론을 모두 포함하며, 최종적으로 csv 데이터를 지정 경로에 저장한다.

7.5.1 데이터 불러오기

각 데이터셋을 불러올 때, dtype은 'object'를 지정하여 데이터 타입을 문자열로 통일하는 전처리를 수행했다. object 데이터 타입은 str과 숫자 데이터가 조합되어 있는 변수를 처리하는 타입이다. 이는 다양한 데이터 타입의 일관성을 유지하고, 이후 데이터 통합 시 발생할 수 있는 타입 오류를 방지하기 위해 수행하였다.

7.5.2 소통 데이터 돌발 데이터 통합

먼저, 소통 데이터와 돌발 데이터는 날짜 데이터와, 링크 아이디 두개가 모두 같은 값들을 소통 데이터가 왼쪽으로, left outer join 하였다. 그러나 소통 데이터는 날짜 형식이 pandas의 타입 스탬프가 아니다. 이에 따라 본 연구에서는 소통 데이터의 생성일과 생성시분을 다음 문자열 변환을 통해 타임스탬프 값으로 변환하였다. 또 돌발 데이터 또한 다음과 같은 변환 과정을 거쳐 타임스탬프 값을 생성하였다.

그 이후, pd.merge 함수를 통해 모든 소통 및 돌발 데이터를 날짜 데이터와 링크 아이디를 프라이머리 키로 조인하였다.

7.5.3 노드링크 데이터 통합

링크 데이터는 링크 아이디를 프라이머리 키로 사용하였다. 링크 데이터는 반드시 필요한 필수 요구 변수이므로 inner join을 사용하였다.

이렇게 생성된 중간 병합 데이터에 시작 지점 노드 아이디를 프라이머리 키로 하여 노드 데이터를 저장하였다. 이 또한 같은 연유로 innder join을 사용하였다.

7.5.4 방재 데이터와 지점 데이터 결합

기상 데이터는 방재 데이터와 지점 데이터를 바로 결합하는 것이 아닌, 방재 데이터와 지점 데이터를 결합한 기상 결합 데이터를 먼저 생성하였다. 프라이머리 키는

지점 코드를 사용하였고, inner과 outer join 모두 같은 결과 데이터를 산출하였다.

7.5.5 기상 결합 데이터 통합

기상 결합 데이터의 지점 데이터를 프라이머리 키로 사용해야 한다. 그러나 중간 병합 데이터는 지점 데이터 열이 존재하지 않는다. 이에 따라 본 연구에서는 각 열마다 지점별 위치 리스트를 생성하였다. 그리고, 중간 병합 데이터 각 열의 위, 경도 데이터와 지점별 위치 리스트의 유클리디안 거리를 산출하고, 최근 거리의 지점 코드를 모아 근접 지점 열을 추가하였다. 그 후 근접 지점 열을 프라이머리 키로 하여 inner join하였다.

7.5.6 혼잡빈도 데이터 통합

혼잡빈도 데이터는 주소 열을 프라이머리키로 하여 중간 병합 데이터를 왼쪽으로 하여 left outer join하였다.

7.5.7 그 외 데이터 결합

인구 데이터는 주소의 시군구 데이터를 프라이머리 키로 하여 left outer join하였다. 운전면허 소지자 데이터는 모든 열에 전체 적용하였다. 그리고 사고다발지점과 단속 카메라 데이터는 위치 데이터를 프라이머리 키로 하여 left outer join하였다.

7.5.8 최종 데이터 산출

모든 과정을 마친 후, 병합이 되지 않아 결측치가 존재하는 열은 모두 동적 대체하였다. 링크 별로 데이터를 그룹화하였고, 중복 제거 후 csv 데이터로 내보냈다.

8장. 실증 연구: 결합 모델 실험 및 평가

8.1 모델 개발

1. 개요

본 절에서는 위에서 고안한 세가지 결합 모델을 실제 학습 및 예측 가능한 인공지능 모델로서 개발하고자 한다. 이를 위해 본 연구에서는 아래의 이론적, 실재적 기법을 활용하였다.

세가지 결합 모델은 Python을 통해 개발되었다. 기존 GRU모델에 결합 모델의 요소들을 추가하기 위하여 머신러닝 프레임워크인 Google의 TensorFlow™를 활용하여 개발되었다. TensorFlow™는 그래픽 최적화를 통한 병렬 연산 처리가 훌륭하고, 또 모델의 세부설정을 변경하기 손쉽기 때문에 사용하였다. 또, 데이터 구성에서 필요한 정규화와 분할 과정은 데이터 전용 프레임워크인 Scikit-learn이 사용되었다. Scikit-learn은 데이터 처리를 위한 다양한 기능이 존재하고, 또 위의 텐서플로우같은 인공지능 프레임워크와 같이 사용할 때 효율이 좋아지기 때문에 사용하였다.

개발한 모델을 기존 방법론들과 비교하기 위하여 객관적인 평가 지표인 MAE, RMSE, MSE, R^2를 사용하였다. MAE(Mean Absolute Error)는 모델의 예측값과 실제값 간의 차이를 평가하는 오차 성능 지표 중 하나로, 각 데이터 포인트에서의 오차 절대값들의 평균을 구하는 방식으로 계산된다. 이는 오차의 방향성을 고려하지

않으므로, 예측이 실제값보다 크거나 작을 때의 차이를 동일하게 반영한다.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

여기서 y_i는 실제값, $\hat{y}_i$는 모델의 예측값, 그리고 n은 데이터의 총 개수이다. MAE는 오차의 크기를 실제 단위 나타낸다. 따라서 MAE는 모델의 예측 성능을 평가할 때 실제값과 예측값 사이의 평균적인 절대 편차를 산출한다. 이는 절대적인 오차 크기를 평가하는 지표로, 비교적 큰 오차나 작은 오차를 동일하게 취급함으로써 평균적인 예측 성능을 평가한다. 따라서 본 연구에서는 모델의 예측 성능을 평가하기 위한 주요 지표로 MAE를 사용하였다.

MSE(Mean Squared Error)는 예측 모델의 성능을 평가하는 데 널리 사용되는 지표로, 실제값과 예측값 간의 차이를 제곱하여 그 평균을 구하는 방식으로 산출된다. 이는 오차의 크기뿐만 아니라, 큰 오차에 대해 더 큰 지표를 부여함으로써 예측의 정확도를 평가한다.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

여기서 y_i는 실제값, $\hat{y}_i$는 모델의 예측값, 그리고 n은 데이터의 총 개수이다. MSE는 오차를 제곱하여 계산하므로, 큰 오차에 더 큰 가중치를 부여한다. 즉, 예측에서 발생한 극단적인 오차에 대해 민감하게 반응함으로써 모델의 성능을 더욱 엄격하게 판단하는 지표로 사용된다. 본 연구에서는 모델의 예측 성능을 종합적으로 평가하기 위해 MSE를 사용하였다.

평균 제곱근 오차(Root Mean Squared Error, RMSE)는 예측 모델의 성능을 평가하는 대표적인 지표로, 평균 제곱 오차(Mean Squared Error, MSE)의 제곱근을 취한 값이다. 이는 예측 오차의 크기를 실제 단위로 다시 변환한다.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

여기서 y_i는 실제값, $\hat{y}_i$는 모델의 예측값, 그리고 n은 데이터의 총 개수이다. RMSE는 MSE와 마찬가지로 예측 오차를 제곱하여 큰 오차에 더 큰 가중치를 부여하는 특성을 가지지만, 제곱근을 취함으로써 그 값을 원래 데이터의 단위로 변환한다. 본 연구에서는 모델의 예측 성능을 평가하기 위한 지표로 RMSE를 사용하였다.

평균 절대 백분율 오차(Mean Absolute Percentage Error, MAPE)는 예측 모델의 성능을 평가하는 지표로, 예측 오차를 실제값에 대한 백분율로 나타내어 상대적인 예측 정확도를 측정하는 데 사용된다. MAPE는 예측값과 실제값 간의 절대 오차를 각각의 실제값으로 나눈 후, 그 값들의 평균을 구하는 방식으로 계산된다.

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

여기서 y_i는 실제값, $\hat{y}_i$는 예측값, 그리고 n은 데이터의 총 개수이다. MAPE는 오차를 백분율로 표현함으로써, 예측 성능을 실제값의 규모와 무관하게 비교할 수 있다. 특히 다양한 스케일을 가진 데이터에 대해 예측 모델의 성능을 비교할 때 유용하게 적용될 수 있다. 다만, MAPE는 실제값이 0에 가까운 경우 오차가 크게 부풀려질 수 있다. 본 연구에서는 예측 모델의 상대적인 성능을 평가하기 위한 지표로 MAPE를 사용하였다.

결정계수(Coefficient of Determination), R^2로 표기되는 이 지표는 회귀 모델의 예측 성능을 평가하는 데 사용된다. R^2는 모델이 종속 변수의 변동성을 얼마나 잘 설명하는지를 나타내며, 모델이 실제값을 얼마나 정확하게 예측했는지를 측정한다.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

여기서 y_i는 실제값, $\hat{y}_i$는 모델의 예측값, $\bar{y}$는 실제값의 평균, 그리고 n은 데이터의 총 개수이다. R^2 값은 0에서 1 사이의 값을 가지며, 1에 가까울수록 모델이 실제값의 변동성을 잘 설명하고 있음을 의미한다. 반대로, 0에 가까운 값은 모델이 거의 설명력을 갖지 못하고 있다는 것을 나타냅니다. 이 지표는 특히 회귀 분석에서 모델의 성능을 정량적으로 평가하는 역할을 하며, 예측값이 실제값과 얼마나 일치하는지를 표현한다. 또한 R^2는 모델이 설명할 수 있는 총 변동성을 비율로 제공하므로, 단순히 오차의 크기를 넘어 모델의 설명력을 평가할 수 있다. 본 연구에서는 제안된 회귀 모델의 설명력을 평가하기 위해 R^2 지표를 사용하였다.

2. 학습 데이터 구성

가. 데이터 분할

효율적인 모델 학습을 위해 학습과 테스트 데이터를 8대 2의 비율로 분할하였다. 특정 검증 데이터까지 포함한다면, 학습 과 검증, 그리고 테스트 데이터를 8 대 1 대 1의 비율로 분할하였다.

데이터 분할은 데이터의 과대, 과소적합을 피하기 위해 필수불가결한 과정이다 **??**.

나. 데이터 정규화

데이터 축소를 데이터를 위해 모두 최소-최대 정규화 하였다. $0 - 1$ 범위로 축소하였다.

3. 결합 모델 구현

각 결합 모델에 필요한 연산 중, 가중치 등의 모델에 직접적으로 영향을 주는 것들은 TensorFlow™의 기능을 활용하였다. 각 결합 모델은 비교 모델들과 구조적

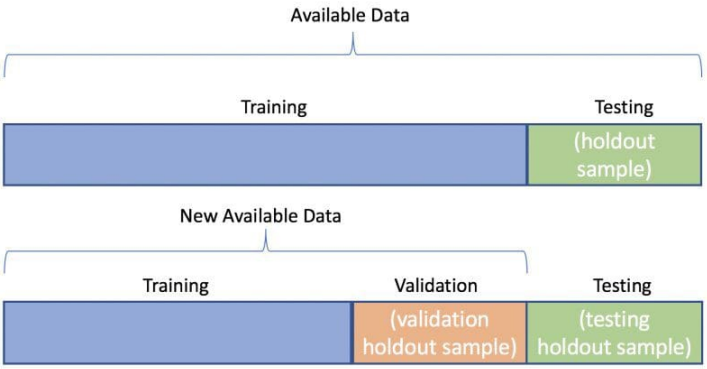

Figure 8.34: 데이터 분할

차이가 있기에 성능에는 차이가 없도록 하이퍼파라미터들을 설정하였다.

가. ANN 기반 충격파 결합 모델

ANN을 활용한 충격파 결합 모델의 코드 중 딥러닝 모델 부분은 다음과 같다.

```
#충격파 강도 ANN 모델
def build_shockwave_model(input_shape):
        model = tf.keras.Sequential([
                tf.keras.layers.Input(shape=input_shape),
                tf.keras.layers.Dense(64, activation='relu'),
                tf.keras.layers.Dense(32, activation='relu'),
                tf.keras.layers.Dense(1)  # 충격파 강도를 예측하는 출력
        ])
        return model

#GRU 연산
class ShockwaveGRUModel(tf.keras.Model):
    def __init__(self, hidden_size, dropout_rate=0.2):
        super(ShockwaveGRUModel, self).__init__()
        self.hidden_size = hidden_size
```

```
        self.gru =
                tf.keras.layers.GRU(hidden_size, return_sequences=True, return_state=True,
                dropout=dropout_rate, recurrent_dropout=dropout_rate)
        self.dense1 = tf.keras.layers.Dense(64, activation='relu')
        self.dropout = tf.keras.layers.Dropout(dropout_rate)
        self.batch_norm = tf.keras.layers.BatchNormalization()
        self.dense2 = tf.keras.layers.Dense(1)
        self.W_shock = tf.Variable(tf.random.normal([1, hidden_size]), trainable=True)

    def call(self, inputs, shock_intensity):
        x = inputs
        gru_out, state = self.gru(x)

        # 충격파 강도를 GRU에 반영
        shock_intensity = tf.expand_dims(shock_intensity, axis=-1)
                # [batch_size, sequence_length, 1]
        shock_intensity = tf.tile(shock_intensity, [1, 1, self.hidden_size])
                # [batch_size, sequence_length, hidden_size]
        shock_effect = tf.matmul(shock_intensity, self.W_shock)
                # [batch_size, sequence_length, hidden_size]
        state = state + tf.reduce_mean(shock_effect, axis=1)
                # [batch_size, hidden_size]

        # 예측 값 계산
        x = self.dense1(state)
        x = self.dropout(x)
        x = self.batch_norm(x)
        output = self.dense2(x)
        return output
```

충격파 강도 산출을 위한 추가 ANN 모델의 설정은 다음과 같다. 은닉 계층은 총 3개이다 **??**. 첫번째 은닉층은 완전연결계층이고, 활성화 함수는 ReLU이고, 크기는 128이다. 두번째 은닉층은 완전연결계층이고, 활성화 함수는 ReLU, 크기는 64이다.

세번째 은닉층은 완전연결계층이고, 활성화 함수는 ReLU, 크기는 32이다.

은닉 계층 번호	종류	활성화 함수	크기
1	Dense	ReLU	128
2	Dense	ReLU	64
3	Dense	ReLU	32

Table 8.28: 별개 ANN의 은닉층 설정

나. 표준편차 기반 충격파 결합 모델

표준편차를 활용한 충격파 결합 모델의 코드 중 데이터 처리와 딥러닝 모델 부분은 다음과 같다.

```
# 교통량 변화율 계산
data['delta_flow'] = data['traffic(Q)'].diff()

# 표준편차 계산
sigma = data['delta_flow'].std()

# 충격파 강도 계산
data['shock_intensity'] = (data['delta_flow'].abs() > sigma).astype(int)

class OptimizedGRUModel(tf.keras.Model):
    def __init__(self, hidden_size, dropout_rate=0.2):
        super(OptimizedGRUModel, self).__init__()
        self.hidden_size = hidden_size
        self.gru = tf.keras.layers.GRU(hidden_size, return_sequences=True,
                return_state=True, dropout=dropout_rate, recurrent_dropout=dropout_rate)
        self.dense1 = tf.keras.layers.Dense(64, activation='relu')
        self.dropout = tf.keras.layers.Dropout(dropout_rate)
```

```
        self.batch_norm = tf.keras.layers.BatchNormalization()
        self.dense2 = tf.keras.layers.Dense(1)
        self.W_shock = tf.Variable(tf.random.normal([1, hidden_size]), trainable=True)

    def call(self, inputs):
        x, shock_intensity = inputs[:, :, :-1], inputs[:, :, -1]
        gru_out, state = self.gru(x)

        # 충격파 강도를 GRU에 반영
        shock_intensity = tf.expand_dims(shock_intensity, axis=-1)
                # [batch_size, sequence_length, 1]
        shock_intensity = tf.tile(shock_intensity, [1, 1, self.hidden_size])
                # [batch_size, sequence_length, hidden_size]
        shock_effect = shock_intensity * self.W_shock
                # [batch_size, sequence_length, hidden_size]
        state = state + tf.reduce_mean(shock_effect, axis=1)

                # [batch_size, hidden_size]

        # Dense 레이어 및 드롭아웃, 배치 정규화 추가
        x = self.dense1(state)
        x = self.dropout(x)
        x = self.batch_norm(x)
        output = self.dense2(x)
        return output
```

데이터 셋에서 표준편차를 numpy의 기능을 통해 계산해서, 충격파 값을 포함하는 새로운 열을 생성하였다. 그 후 모델의 입력 데이터에는 새로운 열을 제외하고 사용하고, 모델의 내부 파라미터로 새로운 열을 입력한다. 이를 통해 결합 모델을 구현할 수 있었다.

다. 이동평균 기반 충격파 결합 모델

이동평균을 활용한 충격파 결합 모델의 코드 중 데이터 처리와 딥러닝 모델 부분은 다음과 같다.

```
# 이동 평균 계산
window_size = 18  # 이동 평균을 계산할 윈도우 크기
data['moving_avg_flow'] = data['speed(u)'].rolling(window=window_size).mean()

# 교통량 변화율 계산
data['delta_flow'] = data['speed(u)'] - data['moving_avg_flow']

# 표준편차를 이용한 임계값 설정
k = 0.7  # 임계값 조정을 위한 상수
sigma = k * data['delta_flow'].std()

# 충격파 강도 계산
data['shock_intensity'] = (data['delta_flow'].abs() > sigma).astype(int)

class MovingAvgGRUModel(tf.keras.Model):
    def __init__(self, hidden_size, dropout_rate=0.2):
        super(MovingAvgGRUModel, self).__init__()
        self.hidden_size = hidden_size
        self.gru = tf.keras.layers.GRU(hidden_size, return_sequences=True,
                return_state=True, dropout=dropout_rate, recurrent_dropout=dropout_rate)
        self.dense1 = tf.keras.layers.Dense(64, activation='relu')
        self.dropout = tf.keras.layers.Dropout(dropout_rate)
        self.batch_norm = tf.keras.layers.BatchNormalization()
        self.dense2 = tf.keras.layers.Dense(1)
        self.W_shock = tf.Variable(tf.random.normal([1]), trainable=True)

    def call(self, inputs):
        x, shock_intensity = inputs[:, :, :-1], inputs[:, :, -1]
        gru_out, state = self.gru(x)
```

```
        # 충격파 강도를 GRU에 반영
        shock_intensity = tf.expand_dims(shock_intensity, axis=-1)
                # [batch_size, sequence_length, 1]
        shock_intensity = tf.tile(shock_intensity, [1, 1, self.hidden_size])
                # [batch_size, sequence_length, hidden_size]
        shock_effect = shock_intensity * self.W_shock
                # [batch_size, sequence_length, hidden_size]
        state = state + tf.reduce_mean(shock_effect, axis=1)
                # [batch_size, hidden_size]

        # Dense 레이어 및 드롭아웃, 배치 정규화 추가
        x = self.dense1(state)
        x = self.dropout(x)
        x = self.batch_norm(x)
        output = self.dense2(x)
        return output
```

표준편차와 마찬가지로 충격파 값을 저장한 새로운 열을 생성하여 사용하였고, 모델 내부 파라미터로 입력하였다.

표준편차의 범위인 window_ size는 10부터 20까지의 값을 모두 에포크 100과 K 값 1의 학습 설정으로 실험해보았을 때 가장 R^2score 값이 높았던 18의 값을 사용하였다. 그리고 K값은 마찬가지로 window_ size 18과 에포크 100일 때 0.5부터 1.5까지의 값까지 실험해보았을 때 가장 R^2score 값이 높았던 0.7의 값을 사용하였다.

라. LWR 기반 충격파 결합 모델

LWR의 충격파 속도를 활용한 충격파 결합 모델의 코드 중 데이터 처리와 딥러닝 모델 부분은 다음과 같다.

```
delta_traffic = np.diff(traffic_data, prepend=traffic_data[0])
delta_speed = np.diff(speed_data, prepend=speed_data[0])
```

```
shock_wave_intensity = np.divide(delta_traffic, delta_speed,
out=np.zeros_like(delta_traffic, dtype=float), where=delta_speed != 0)

class ShockwaveGRU(tf.keras.layers.Layer):
    def __init__(self, units, **kwargs):
        super(ShockwaveGRU, self).__init__(**kwargs)
        self.units = units
        self.gru = tf.keras.layers.GRU(units, return_state=True)

    def build(self, input_shape):
        # W_shock은 스칼라 형태의 가중치로 초기화
        self.W_shock = self.add_weight(shape=(1, 1),
                                       initializer='random_normal',
                                       trainable=True, name='W_shock')
        super(ShockwaveGRU, self).build(input_shape)

    def call(self, inputs, states):
        shock_intensity = inputs[..., -1:]
        shock_contribution = shock_intensity * self.W_shock
        modulated_input = tf.concat([inputs[..., :-1], shock_contribution], axis=-1)
        output, state = self.gru(modulated_input, initial_state=states)
        return output, state
```

numpy 기능을 통해 충격파 강도 열을 생성하였다.

마. 뉴웰의 차량추종모델 결합 모델

뉴웰의 차량추종모델 결합 모델의 데이터 처리와 딥러닝 모델 부분은 다음과 같다.

```
data['delta_x'] = data['speed(u)'].diff()

K = 3  # 3 스텝 뒤에 반응

```

```python
# 입력 데이터 변화량을 K타임 스텝 뒤로 이동하여 반응
data['delta_x_shifted'] = data['delta_x'].shift(K)

class NewellGRUModel(tf.keras.Model):
    def __init__(self, hidden_size, dropout_rate=0.2):
        super(NewellGRUModel, self).__init__()
        self.hidden_size = hidden_size
        self.gru = tf.keras.layers.GRU(hidden_size, return_sequences=True,
                return_state=True, dropout=dropout_rate, recurrent_dropout=dropout_rate)
        self.dense1 = tf.keras.layers.Dense(64, activation='relu')
        self.dropout = tf.keras.layers.Dropout(dropout_rate)
        self.batch_norm = tf.keras.layers.BatchNormalization()
        self.dense2 = tf.keras.layers.Dense(1)
        self.T = tf.Variable(tf.random.normal([1]), trainable=True)  # 변화량의 가중치

    def call(self, inputs):
        x, delta_x_shifted = inputs[:, :, :-1], inputs[:, :, -1]
        gru_out, state = self.gru(x)

        # 리셋 게이트와 업데이트 게이트에 변화량 반영
        delta_x_effect = self.T * tf.expand_dims(delta_x_shifted, axis=-1)
                # [batch_size, sequence_length, 1]
        delta_x_effect = tf.tile(delta_x_effect, [1, 1, self.hidden_size])
                # [batch_size, sequence_length, hidden_size]

        # 은닉 상태에 변화량의 효과 반영
        state = state + tf.reduce_mean(delta_x_effect, axis=1)
                # [batch_size, hidden_size]

        # Dense 레이어 및 드롭아웃, 배치 정규화 추가
        x = self.dense1(state)
        x = self.dropout(x)
        x = self.batch_norm(x)
        output = self.dense2(x)
```

```
        return output
```

시간 지연의 정도인 K 스텝 값은 1부터 5까지의 값 중 가장 MAPE값이 높았던 3을 사용하였다.

바. 다상교통이론 결합 모델

뉴웰의 차량추종모델 결합 모델의 딥러닝 모델 부분은 다음과 같다.

```
class MultiPhaseGRUCell(tf.keras.layers.GRUCell):
    def __init__(self, units, state_matrix_size, **kwargs, P_m, H_m):
        super(MultiPhaseGRUCell, self).__init__(units, **kwargs)
        self.state_matrix_size = state_matrix_size
        self.P = P_m # P_m은 미리 설정된 상태 영향 행렬, 형태는 shape=(state_matrix_size,)
        self.H = H_m # H_m은 미리 설정된 상태 전이 영향 행렬,
                # 형태는 shape=(state_matrix_size, state_matrix_size)

    def call(self, inputs, states):
        prev_hidden_state = states[0]
        S = tf.nn.softmax(inputs)
                #inputs은 외부 ANN모델의 출력, 소프트맥스는 거치지 않은 상태로 받는다.
        current_state = tf.argmax(S, axis=-1)
        P_current = tf.gather(self.P, current_state)
        prev_state = tf.argmax(tf.nn.softmax(prev_hidden_state), axis=-1)
        H_transition = tf.gather(self.H[prev_state], current_state)
        output, new_states = super(MultiPhaseGRUCell, self).call(inputs, states)
        updated_hidden_state = output + P_current + H_transition
        return updated_hidden_state, [updated_hidden_state]

def build_multi_phase_gru_model(input_shape, units, state_matrix_size):
    inputs = tf.keras.Input(shape=input_shape)
    gru_layer = tf.keras.layers.RNN(MultiPhaseGRUCell(units, state_matrix_size),
        return_sequences=True)(inputs)
    outputs = tf.keras.layers.Dense(1, activation='linear')(gru_layer)
```

```
    model = tf.keras.Model(inputs, outputs)
    model.compile(optimizer='adam', loss='mse')
    return model
```

세가지의 상태 설명 행렬들에 대한 설정은 코드 내 주석과 같다.

4. 공통 모델 설정

세가지 결합 모델의 모델의 구조에 관한 설정이다. 이 설정들은 후에 성능 비교를 위한 기존 딥러닝적 방법들에도 동일하게 적용된다.

가. 입력 변수 설정

전처리를 완료한 분석 데이터의 변수들은 딥러닝 모델의 입력변수로 활용되었다. 입력 변수 중 비-숫자형 변수들은 원-핫 인코딩을 통해 숫자형으로 변환되고(One-hot Encoding), 숫자형 변수들은 정규화(Normalization)된 값으로 사용하였다.

나. 타겟 데이터

교통량 예측 모델이기 때문에, 타겟 변수로는 다음 행의 교통량 데이터를 사용하였다.

다. 은닉 상태 hidden size

hidden size는 모두 64로 통일하였다.

라. 옵티마이저

옵티마이저는 Adam을 사용하였고, 학습률은 0.001를 사용하였다.

마. 베치 크기

베치크기는 32를 사용하였다.

바. 드롭아웃

드롭아웃 비율은 0.2를 사용하였다.

사. 손실함수

mse를 사용하였다.

아. 에포크

에포크 크기는 1000을 사용하였다.

자. 초기화

He초기화 등의 특별한 초기화 기법들이 존재하지만, 본 연구에서는 기본적인 랜덤 초기화를 사용하였다.

차. 은닉층 수

은닉층은 1개만 사용하였다.

8.2 실험 및 평가

8.2.1 학습 설정

VDS식별 코드로 구분된 각 데이터 셋들을 모두 학습하기 위해 연속학습 기법을 사용하였다. 연속학습(Continual Learning)은 딥러닝 모델이 새로운 데이터를

항목	값	항목	값
은닉층 수	1	드롭아웃	0.2
은닉 크기	64	배치 크기	32
타겟 변수	교통량	손실함수	MSE
옵티마이저	Adam	에포크	1000
학습률	0.001	초기화	Random

Table 8.29: 결합 모델 공통 설정

기반으로 지속적으로 학습하는 방법을 뜻한다. 기존의 딥러닝 모델은 대규모의 데이터들로 학습하고, 해당 데이터셋을 기반으로 한 일반적 패턴을 학습한다. 그러나 실제 데이터에서는 새로운 데이터가 지속적으로 추가되고, 이 데이터들은 기존의 학습 데이터와는 다를 가능성이 있다. 연속학습은 이러한 기존 딥러닝의 문제점을 보완하고자 등장하였다. 새로운 데이터에 대해 지속적으로 학습하고, 학습한 지식을 점진적으로 확장해나간다.

본 연구에서는 연속학습을 구현하기 위해 다음과 같은 방법을 사용했다. 먼저, 각 데이터 셋을 순차적으로 학습한다. 이때, 초기 가중치 설정을 전 데이터 셋의 학습 결과로 세팅한다. 이를 통해 각 데이터를 하나의 데이터처럼 취급하며 데이터 과적합과 망각을 방지할 수 있다.

이렇게 학습이 완료된 후, 연구의 목적에 따라 예측 및 실험을 진행하기 위해서는 학습된 모델을 저장할 적절한 방법이 필요하다. 본 연구에서는 가장 간편하고, 별다른 설정이 불필요한 Python의 Pickle이라는 기능을 이용하였다. Pickle은 파이썬의 List와 Dictionary같은 자료구조를 파일 형태로 보존할 수 있게 도와주는 패키지이다. 이 기능을 통해 각 모델의 가중치 등 설정을 저장하였다.

입력 변수는 교통량을 제외한 유의 변수를 사용하였다. 입력 변수에 교통량을

넣은 뒤 학습시킨 결과, 모든 모델의 성능이 거의 비슷해 비교가 불가능 하였기 때문이다. 또 실제 교통 예측 환경에서는 전 시간대의 교통량 정보가 제공되었다면, 본 연구에서 개발한 교통 예측 모델의 효용성이 무의미해지기 때문이다. 따라서 자기회귀 예측법이 아닌 외부 요인 기반 학습법을 통해 학습 및 예측을 진행하였다. 그러나, 그래프 분석 기법과 같이 자기 회귀 예측법만을 허용하는 모델에서는 자기 회귀적 예측법을 사용하여 비교하였다.

8.2.2 딥러닝 기반 기법들과의 비교

딥러닝 기반 기법들은 인공 신경망을 기반으로 하는 ANN(Artificial Neural Network), BRR(Bayesian Ridge Regression), DBN(Deep Belief Network), GBM(Gradient Boosting Algorithm), KNN-R(KNN Regression), LR(Linear Regression), LWL(Locally Weighted Linear Regression), RF(Random Forest), RBF(Radial Basis Function), SAE(Stacked AutoEncoder), SVR(Support Vector Regression)과 기존 순환 신경망인 GRU을 개발하여 오차와 정확도를 비교하였다. 딥러닝 기반 모델들은 구조가 근본적으로 동일 하기 때문에 결합 모델의 모델 설정을 모두 동일하게 적용하였다.

또한 모두 연속학습을 적용하였고, 1, 4, 7, 10월의 각 한가지 지점의 데이터를 제외한 모든 데이터 셋을 학습 시켰다. 그 후 피클로 저장된 모델로 각 달의 제외된 지점의 데이터를 예측하였다. 성능 지표와 데이터의 정확도를 위하여 제외된 지점의 각 예측 성능 지표를 평균내어 사용하였다. 제외할 지점은 Python의 Random기능을 사용하였다. 다른 기법들과의 비교는 각 데이터 셋의 100% 를 사용하여 거시적 관점에서의 예측만을 비교하였다.

GBM은 n_ estimators를 1000으로, 최대 깊이(Max_ Depth)는 5로 설정하였다. KNN-R은 에포크가 존재하지 않기에 적용하지 않았다. 또 KNN-R의 n-neighbors 값은 같은 개념인 배치 크기 32와 동일하게 설정하였다. 선형회귀는 에포크가 존재하지 않아 적용하지 않았다. LWL 모델은 폭 0.5의 가우시안 커널과 유클리드 제곱

거리를 사용하여 가중치를 계산하였다. RBF는 gamma값을 1로 설정하였다. SAE는 인코더와 디코더 각 3개씩, 유닛 수는 64, 32, 16로 설정하였다. SVR모델은 RBF 커널을 사용하였다. 순환 신경망 시퀀스(Sequence) 모델은 시퀀스 길이를 10으로 설정하였다.

가. 전체 데이터 예측

9장. Case Study

10장. 결론 및 향후 연구 과제

부록

A 테스트 코드

B 부록 두번째 제목

필요한 경우 색인(index)을 작성한다.