Time-series Extreme Event Forecasting with Neural Networks at Uber

0. Abstract

- 변동이 큰 시기(휴일 등) 동안의 정확한 시계열 예측은 이상탐지, 자원분배 최적화, 예산 계획 등의 많은 분야에서 중요하다.
- 우버의 경우, 특정 이벤트 상황에서의 수요를 정확하게 예측하는 것이 더욱 효율적인 드라이버 할당으로 이어져, 손님들이 기다리는 시간을 줄여준다.
- 이러한 문제를 해결하는 SOTA method들은 보통 단변량 예측모델(Holt-Winters 등)과 ML method(랜덤포레스트 등)를 결합한 것에 의존해왔다.
- 그러나, 이러한 시스템은 튜닝이 어렵고 확장성이 떨어지며 외부 변수를 추가하기 어렵다.
- 최근의 LSTM 모델에 영감을 받아, 우리는 end-to-end RNN 구조를 제안한다.
    - 현재 우버에서 사용하는 SOTA 예측 모델보다 나은 성능을 보인다.
    - 시계열 예측 대회에서 사용되는 M3 데이터셋에 상대적으로 나은 일반화 성능을 보인다.

1. Introduction

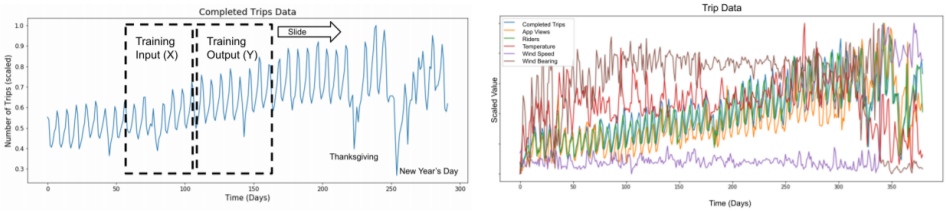
- 변동이 큰 시기의 시계열 예측이 어려운 이유는 다양한 외부 변수에 영향을 받기 때문이다.
    - 계절, 인구 성장세, 운전자 인센티브 정책 등 마케팅의 변화 등...
- 외부 변수를 반영하는 QRF 기반의 기존 모델은 재학습 빈도가 높아 확장성이 떨어진다.
- 고전적인 모델들은 계절성과 다른 파라미터를 세팅하기 위해 수동 튜닝이 필요하다.
    - 시계열 모델의 경우, 외부 변수를 포함할때 차원의 저주와 잦은 재학습으로 고통받는다.
    - 외부 변수를 효과적으로 다루기 위해 단변량 모델링과 ml 모델을 결합하여 잔차를 통제하려는 시도도 있었으나, 이 역시 튜닝이 어렵고 수동으로 feature를 만들어줘야 하는 점, 그리고 잦은 재학습이 요구되었다.
- 최근, LSTM에 기반한 시계열 모델링은 여러 장점으로 주목받고 있다.
    - end-to-end modeling
    - 외부 변수 반영이 쉬움
    - 자동 feature 생성 능력
    - 다량의 데이터를 제공해주기만 하면, 복잡한 비선형의 feature interaction을 모델링할 수 있는데, 이는 complex extreme events를 모델링함에 있어 매우 중요하다.
- Contribution
    - 새로운 LSTM 기반의 구조와 이질적인 시계열 데이터를 사용하여 single model을 학습시키는 방법을 제안함
    - 모델의 일반화 성능과 확장성을 실험으로 증명함

2. Background

- 대형 이벤트(Extreme event)를 예측하는 것은 다양한 분야에서 이미 중요한 주제였다.
    - 전기 수요의 고점(peak) 예측
    - 교통 체증의 심각도
    - 차량 공유의 유동적 가격책정
- 이런 종류의 문제를 다루는, EVT라는 통계학의 한 부류도 존재한다. (Extreme value Theory)
- peak를 예측하는 문제를 해결하기 위해, 단변량 시계열과 ml 접근들이 제시되었다.
    - 단변량 시계열 접근은 일시적 도메인을 직접 모델링하지만, 잦은 재학습이 필요하다.
    - ml 접근은 단변량 시계열과 결합하여 덩치가 큰 2-step process로 만들어진다.
- LSTM은 기존의 시계열 접근방법과 같이 일시적 도메인을 모델링할 수 잇으면서 동시에 비선형적 feature interaction과 잔차를 모델링할 수 있다.
    - 튜닝하지 않은 vanilla LSTM은 우리의 baseline 모델보다 성능이 낮았다.
    - 따라서, 우리는 feature 추출을 위해 autoencoder를 활용하는 새로운 구조를 제시한다.

3. Data

- 우버는 수백개 도시의 운전자와 사용자의 익명화된 접촉 정보를 가지고 있다.
    - 그러나 이렇게 많은 정보를 가지고 있다 해도, 신규 도시와 특별한 이벤트의 데이터는 부족하기 때문에 문제가 발생한다.
    - 이러한 정보 부족을 해결(우회)하기 위해, 추가 feature를 사용한다.
        - 기후 정보 (풍속, 기온 등)
        - city level (사용량, 유저 수, 지역별 휴일)
- 데이터셋 생성
    - 하나의 시계열에 대해, **window X와 Y를 sliding**하면서 생성
        - X : input, 과거 정보
        - Y : output, 예측해야 하는 미래 정보
        - X와 Y는 (batch, time, features)로 구성됨
    - 뉴럴넷은 스케일링 되지 않은 데이터에 민감하므로, 모든 mini-batch에 대해 **정규화**를 적용했다.
    - 또한, de-seasoning과 반대되는 **de-trending**이 더 나은 결과를 보여준다는 점을 찾아냈다.



(a) Creating an input for the model requires two sliding windows for x and for y

(b) A scaled sample input to our model

Figure 1. Real-world time-series examples.

4. Modeling

4.1 Uncertainty estimation

- 현실에서의 대형 이벤트는 확률적으로 발생하며, 따라서 시계열 예측 뉴럴넷에서의 강건한 불확실성 추정은 매우 중요하다.
- 베이지안부터 부트스트랩 이론에 기초한 것까지, 불확실성 추정을 위한 다양한 접근이 있었다.
- 우리는 부트스트랩과 베이지안 접근은 결합하여, 단순하고 강건하면서도 짧은 불확실성 구간으로 좋은 coverage를 보일 것이다.

- 이러한 접근의 적용은 매우 간단하며 실용적이다. (listing 1을 참고)
  - 랜덤한 dropout을 적용한 모델 결과값을 100개 뽑아서 그것들의 평균과 분산으로 불확실성 추정
- 그림 2의 a와 b는 각각 불확실성 도출과 모델의 기저를 묘사한다.
  - 모델의 불확실성은 input과output이 동일 (X → LSTM layers → X)
  - 예측의 불확실성은 input과output이 다름 (X → LSTM layers → Y)
- 보다 구체적인 내용에 대해서는 더 긴 버전의 paper를 위해 여기에는 쓰지 않는다.

Listing 1. Practical implementation of estimating the uncertainty bound

```
vals = []
for r in range(100):
    vals.append(model.eval(input,
                           dropout = random(0,1)))
mean = np.mean(vals)
var = np.var(vals)
```

$$u_y = \sqrt{(y - \hat{y})^2}$$

$$u_y = \sqrt{\varepsilon_y^2}$$

Uncertainty of an input given the model

$$y = \hat{y} + \varepsilon_{X|M,\theta} + \varepsilon_{Y|X,M,\theta}$$

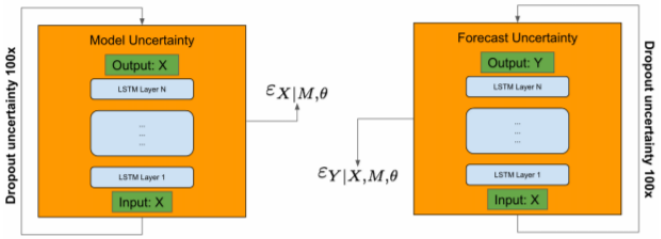
Uncertainty of the forecast given model

$$u_y^2 = \left(\varepsilon_{X|M,\theta} + \varepsilon_{Y|X,M,\theta}\right)^2$$

$$u_y^2 = \left(\varepsilon_{X|M,\theta}^2 + \varepsilon_{Y|X,M,\theta}^2 + 2(\varepsilon_{X|M,\theta}\varepsilon_{Y|X,M,\theta})\right)$$

$$u_y = \sqrt{\varepsilon_{X|M,\theta}^2 + \varepsilon_{Y|X,M,\theta}^2 + 2(\varepsilon_{X|M,\theta}\varepsilon_{Y|X,M,\theta})}$$

(a) Model and forecast uncertainty derivation



(b) Model uncertainty is estimated via the architecture on the left while the forecast uncertainty is estimated via the architecture on the right.

Figure 2. Model and forecast uncertainty

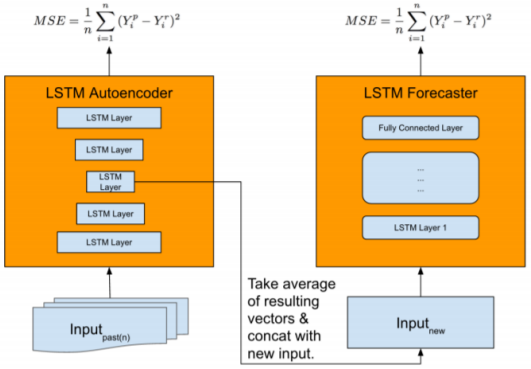
4.2. Heterogeneous forecasting with a single model

- 수백만개의 지표 각각의 시계열 모델을 학습시키는 것은 불가능하다.
  - 따라서, 우리는 heterogeneous forcasting을 위한 single model을 제안한다.
  - (다양한 지표의 예측을 하나의 모델로 처리한다는 뜻으로 해석됨. input으로 라벨을 붙인다던지...)
  - 그림 3의 b에서는 복잡한 시계열의 변동성을 포착하는 데에 필수적인 자동 feature 추출 과정을 보여준다.
    - 이는 과거의 표준화된 feature 추출과 상반되는 것으로, 당시의 feature는 수동으로 개발되었다. (그림3의 a)
  - feature 벡터는 앙상블 기법(평균 등)을 통해 집계되고, 최종 벡터는 새로운 외부 input과 concat되어 LSTM 예측기에 예측을 위해 입력된다.

- 이러한 기법을 활용하여 기존 대비 14%의 성능 개선을 이끌어냈다.

Feature	Description
Mean	Mean.
Var	Variance.
ACF1	First order of autocorrelation.
Trend	Strength of trend.
Linearity	Strength of linearity.
Curvature	Strength of curvature
Season	Strength of seasonality.
Peak	Strength of peaks.
Trough	Strength of trough.
Entropy	Spectral entropy.
Lumpiness	Changing variance in remainder.
Spikiness	Strength of spikiness
Lshift	Level shift using rolling window.
Vchange	Variance change.
Fspots	Flat spots using disretization.
Cpoints	The number of crossing points.
KLscore	Kullback-Leibler score.
Change.idx	Index of the maximum KL score.

(a) Classical time-series features that are manually derived (Hyndman et al., 2015).



(b) An auto-encoder can provide a powerful feature extraction used for priming the Neural Network.

Figure 3. Single model heterogeneous forecast.

5. Results

- 학습은 AWS GPU instance에서 tensorflow로 수행됨
- 예측 오차 지표로는 SMAPE를 사용함

5.1 Special Event Forecasting Accuracy

- 5년간의 미국 내 운행 데이터를 사용, 주요 휴일의 수요를 예측함
- 불확실성은 변동계수로 측정됨(Coefficient of Variation, 표준편차를 산술평균으로 나눈 것)
    - 크리스마스의 불확실성이 가장 큰 것으로 확인됨
- 2% ~ 18%의 예측 정확도 개선이 확인됨

5.2 General Time-Series Forecasting Accuracy

- 일반적인 시계열의 예측 정확도
    - SMAPE의 평균이 32에서 26으로 개선
- M3 데이터셋으로 일반화 성능을 확인

6. Discussion

- 시계열의 수가 많고, 시계열의 길이가 길며, 시계열간의 연관성(상관계수)이 높을 때에만 뉴럴넷이 올바른 선택일 수 있다.
    - 그렇지 않다면 기존의 고전적 모델링이 더 높은 성능을 낼 수도 있다.