DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks

ICML 2017

V. Flunkert, D. Salinas, J. Gasthaus
Amazon Development Center

Paper Review by Doyeon Yoon

February 25, 2019

Contents

Introduction

Related Work

Model
- Architecture
- Likelihood model
- Training
- Scale handling
- Prediction

Experiments

Conclusion

Contents

Introduction

Related Work

Model
- Architecture
- Likelihood model
- Training
- Scale handling
- Prediction

Experiments

Conclusion

Introduction : Motivation

Classical forecasting model

- ARIMA (Autoregressive Integrated Moving Average)
- Exponential smoothing

A New type of forecasting problem

- Forecasting thousands or millions of related time series

Introduction : Motivation

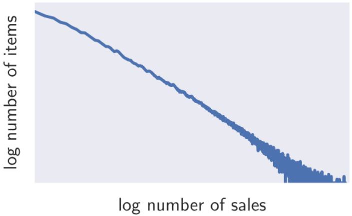


Figure 1: Log-log histogram of the number of items versus number of sales for the 500K time series of ec, showing the scale-free nature (approximately straight line) present in the ec dataset (axis labels omitted due to the non-public nature of the data).

There are two problems

- √ Magnitudes of the time series differ widely
- √ Distribution of the magnitudes is strongly skewed

Divide the data set into sub-groups of time series?

- √ Each velocity sub-group would have a similar skew
- √ Velocities will be vastly different within each group

Introduction : Contribution

1. Propose **RNN architecture** for **probabilistic forecasting**
   - √ Incorporating a Negative Binomial likelihood for count data
   - √ Special treatment for the case when the magnitudes of the time series vary widely
2. **Demonstrate** this model produces accurate probabilistic forecasts.
   - √ across a range of input characteristic, on several real-world data sets.
   - √ in contrast to common belief

Introduction : Key Advantages

- √ Minimal feature engineering
- √ Probabilistic forecasting
- √ Forecast for new items or no history
  - a case where traditional single-item forecasting methods fail
- √ Not assume Gaussian noise
  - allowing the user to choose one that is appropriate for the statistical properties of the data.

Contents

Introduction

Related Work

Model
- Architecture
- Likelihood model
- Training
- Scale handling
- Prediction

Experiments

Conclusion

Related Work

Forecasting individual time series

- ARIMA models
- exponential smoothing methods

Demand forecasting domain

- data preprocessing methods often do not alleviate these conditions
- incorporated more suitable likelihood functions (zero-inflated Poisson, Negative binomial,)

Sharing information across time series

- can improve the forecast accuracy
  - ⇒ difficult to accomplish in practice
  - ⇒ hierarchical structure

Contents

Introduction

Related Work

Model

- Architecture
- Likelihood model
- Training
- Scale handling
- Prediction

Experiments

Conclusion

Model : Notation(Input layer)

Definition

$$z_{i,1:T} := [z_{i,1}, ..., z_{i,t_o-1}, z_{i,t_o}, ..., z_{i,T}]$$

Definition (Covariates)

$covariates$ are assumed to be known for all time points, denoted by

$$x_{i,1:T} := [x_{i,1}, x_{i,2}, ..., x_{i,T}]$$

Model : Architecture

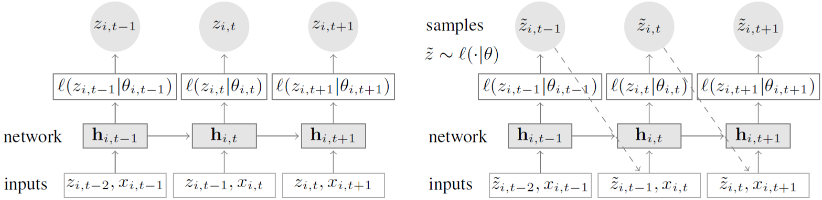


Figure: Training(Left) and Prediction(Right)

Goal is to model the conditional distribution $P(z_{i,t_o:T}|z_{i,1:t_o-1}, x_{i,1:T})$ of the future of each time series $z_{i,t_o:T}$ given its past $z_{i,1:t_o-1}$

Model : Likelihood model
Two choices

Gaussian likelihood : real-valued data

- $\ell_G(z|\mu,\sigma) = (2\pi\sigma^2)^{-\frac{1}{2}} exp(-(z-\mu)^2/(2\sigma^2))$
- $\mu(h_{i,t}) = w_\mu^T h_{i,t} + b_\mu$
- $\sigma(h_{i,t}) = log(1 + exp(w_\sigma^T h_{i,t} + b_\sigma))$

Negative Binomial likelihood : positive count data

- $\ell_{NB}(z|\mu,\alpha) = \dfrac{\Gamma(z+\frac{1}{\alpha})}{\Gamma(z+1)\Gamma(\frac{1}{\alpha})} \left(\dfrac{1}{1+\alpha\mu}\right)^{\frac{1}{\alpha}} \left(\dfrac{\alpha\mu}{1+\alpha\mu}\right)^{z}$
- $\mu(h_{i,t}) = log(1 + exp(w_\mu^T h_{i,t} + b_\mu))$
- $\alpha(h_{i,t}) = log(1 + exp(w_\alpha^T h_{i,t} + b_\alpha))$

√ $\alpha \in \mathbb{R}^+$: shape parameter

Model : Likelihood model
Binomial distribution VS Negative Binomial Distribution

Binomial : $X \sim B(n, p)$(trial n, probability p)

- Count the number of **success**
- Fixed number of **trial**
- $np \to \lambda, n \to \infty$

Model : Likelihood model

Binomial distribution VS Negative Binomial Distribution

Binomial : $X \sim B(n, p)$(trial n, probability p)

- Count the number of **success**
- Fixed number of **trial**
- $np \to \lambda, n \to \infty$

$\Rightarrow$ **Poisson distribution :** $X \sim Pois(\lambda)$

- Given number of events on a fixed interval.

Negative Binomial : $X \sim NB(\mu, \alpha)$(number of success μ, probability α)

- Count the number of **trial**
- Fixed number of **success**

Model : Likelihood model

Binomial distribution VS Negative Binomial Distribution

Binomial : $X \sim B(n, p)$(trial n, probability p)

- Count the number of **success**
- Fixed number of **trial**
- $np \to \lambda, n \to \infty$

  $\Rightarrow$ **Poisson distribution :** $X \sim Pois(\lambda)$

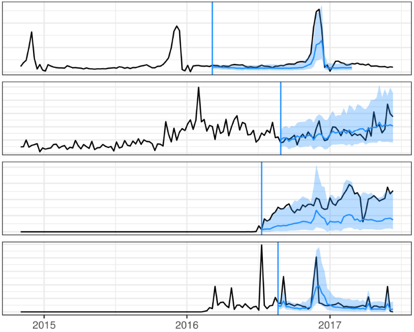
  - Given number of events on a fixed interval.

Negative Binomial : $X \sim NB(\mu, \alpha)$(number of success μ, probability α)

- Count the number of **trial**
- Fixed number of **success**

Related works : Estimating negative binomial demand for retail inventory management with unobservable lost sales(1996)

Generate multiple training instances



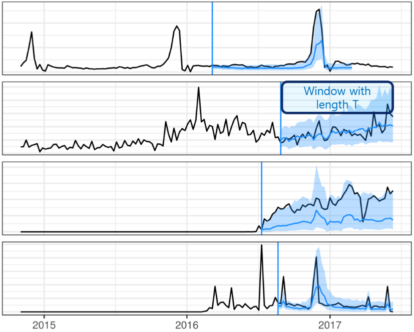
Input

- Past values : $z_1, z_2, ..., z_{t_o - 1}$
- Covariates : $x_1, x_2, ..., x_T$
  - !! Length of $x_{1,T} = t_o + T$

Output

- Distribution $P(z_{t_o}, z_{t_o+1}, ..., z_{t_o+T})$

Generate multiple training instances



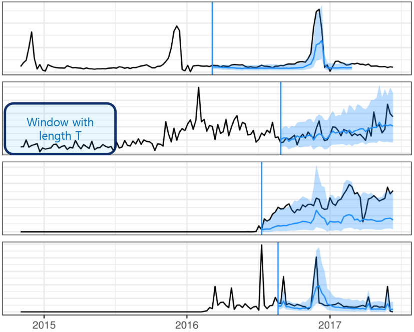
Input

- Past values : $z_1, z_2, ..., z_{t_o-1}$
- Covariates : $x_1, x_2, ..., x_T$
  - !! Length of $x_{1,T} = t_o + T$

Output

- Distribution $P(z_{t_o}, z_{t_o+1}, ..., z_{t_o+T})$

Generate multiple training instances



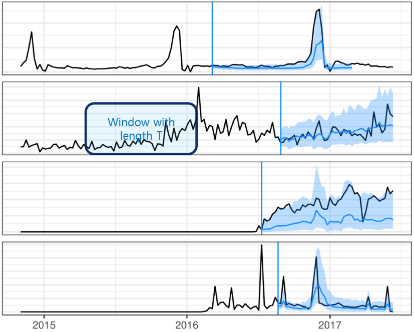
Input

- Past values : $z_1, z_2, \ldots, z_{t_o - 1}$
- Covariates : $x_1, x_2, \ldots, x_T$
  - !! Length of $x_{1,T} = t_o + T$

Output

- Distribution $P(z_{t_o}, z_{t_o+1}, \ldots, z_{t_o+T})$

Generate multiple training instances



Input

- Past values : $z_1, z_2, ..., z_{t_o - 1}$
- Covariates : $x_1, x_2, ..., x_T$
  - !! Length of $x_{1,T} = t_o + T$

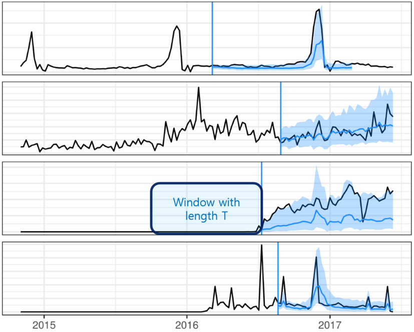
Output

- Distribution $P(z_{t_o}, z_{t_o+1}, ..., z_{t_o+T})$

Model : Training

multiple training by window

Generate multiple training instances



Input

- Past values : $z_1, z_2, \ldots, z_{t_o-1}$
- Covariates : $x_1, x_2, \ldots, x_T$
  - !! Length of $x_{1,T} = t_o + T$

Output

- Distribution $P(z_{t_o}, z_{t_o+1}, \ldots, z_{t_o+T})$

Log Maximum Likelihood :

$$\mathcal{L} = \sum_{i=1}^{N} \sum_{t=t_0}^{T} log\ell(z_{i,t}|\theta(h_{i,t}))$$

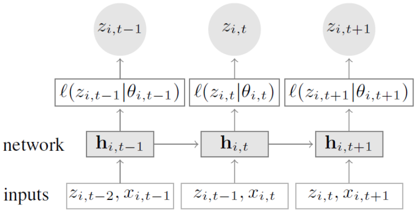


Figure: Encoder

$Likelihood\ function\ \mathcal{L}(\theta|x)$ is defined by probability of x when the parameter θ is given.

$$\Rightarrow \mathcal{L}(\theta|x) = P(x|\theta)$$

Model : Scale handling

Nonlinearity of the networks

- Adjust before and after input / output so as not to be affected by nonlinearity of the networks.

ex Negative binomial distribution

- $\mu = v_i log(1 + exp(o_\mu)), \alpha = log(1 + exp(o_\alpha))/\sqrt{v_i}$
- $Scale\ v_i = 1 + \frac{1}{t_0} \sum_{t=1}^{t_0} z_{i,t}$

Pick training instance uniformly $\Rightarrow$ underfitting

- The probability of selecting a window with scale v_i is proportional to v_i

Model : Prediction

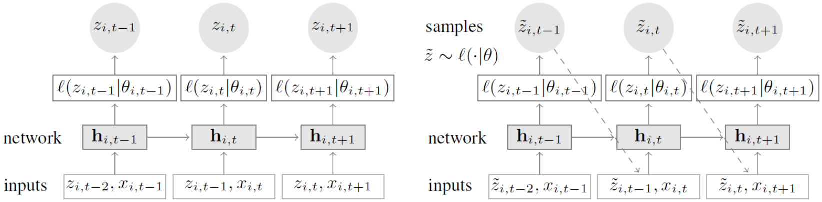


Figure: Training(Left) and Prediction(Right)

Model : Prediction

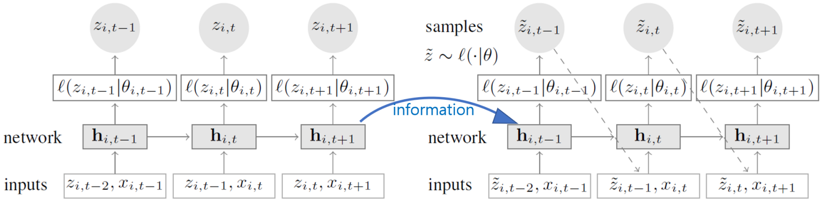


Figure: Training(Left) and Prediction(Right)

Model : Prediction

Generating one sampling trace

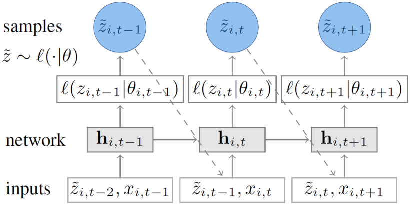


Figure: Decoder

Model : Prediction

Generating one sampling trace

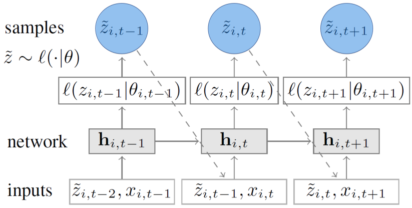
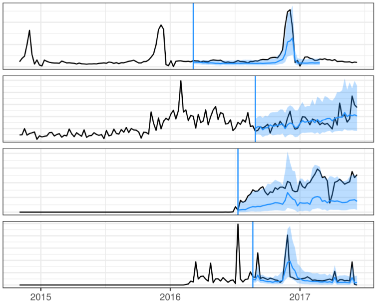


Figure: Decoder

Representing the joint predicted distribution



Contents

Introduction

Related Work

Model
- Architecture
- Likelihood model
- Training
- Scale handling
- Prediction

Experiments

Conclusion

Experiments

Environments

- **laptop:** 1AWS p2.xlarge instance:4CPU + 1GPU
- **framework:** MXNet
- **network:** LSTM

Datasets

	parts	electricity	traffic	ec-sub	ec
# time series	1046	370	963	39700	534884
time granularity	month	hourly	hourly	week	week
domain	$\mathbb{N}$	$\mathbb{R}^+$	$[0, 1]$	$\mathbb{N}$	$\mathbb{N}$
encoder length	8	168	168	52	52
decoder length	8	24	24	52	52
# training examples	35K	500K	500K	2M	2M
item input embedding dimension	1046	370	963	5	5
item output embedding dimension	1	20	20	20	20
batch size	64	64	64	512	512
learning rate	1e-3	1e-3	1e-3	5e-3	5e-3
# LSTM layers	3	3	3	3	3
# LSTM nodes	40	40	40	120	120
running time	5min	7h	3h	3h	10h

Example time series of ec

- Blue line : The p50
- Shaded : The 80% confidence interval

Experiments : comparison

ρ-risk : normalized sum of ρ-quantile losses

	0.5-risk				0.9-risk				average
					parts				
(L, S)	$(0, 1)$	$(2, 1)$	$(0, 8)$	all(8)	$(0, 1)$	$(2, 1)$	$(0, 8)$	all(8)	average
`Snyder` (baseline)	1.00	1.00	1.00	**1.00**	1.00	1.00	**1.00**	1.00	1.00
`rnn-gaussian`	1.17	1.49	1.15	1.56	1.02	0.98	1.12	1.04	1.19
`rnn-negbin`	**0.95**	**0.91**	0.95	**1.00**	1.10	**0.95**	1.06	0.99	0.99
`DeepAR`	0.98	**0.91**	**0.91**	1.01	**0.90**	**0.95**	**0.96**	**0.94**	**0.94**
					ec-sub				
(L, S)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	average
`ISSM` (baseline)	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00
`rnn-gaussian`	1.03	1.19	1.24	0.85	0.91	1.74	2.09	0.67	1.21
`rnn-negbin`	0.90	0.98	1.11	0.85	1.23	1.67	1.83	0.78	1.17
`DeepAR`	**0.64**	**0.74**	**0.93**	**0.73**	**0.71**	**0.81**	1.03	**0.57**	**0.77**
					ec				
(L, S)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	average
`ISSM` (baseline)	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00
`rnn-gaussian`	0.89	0.91	0.94	1.14	0.90	1.15	1.23	**0.90**	1.01
`rnn-negbin`	0.66	0.71	**0.86**	**0.92**	0.85	1.12	1.33	0.98	0.93
`DeepAR`	**0.59**	**0.68**	0.99	0.98	**0.76**	**0.88**	**1.00**	0.91	**0.85**

Figure: Accuracy metrics relative to the strongest previously published method

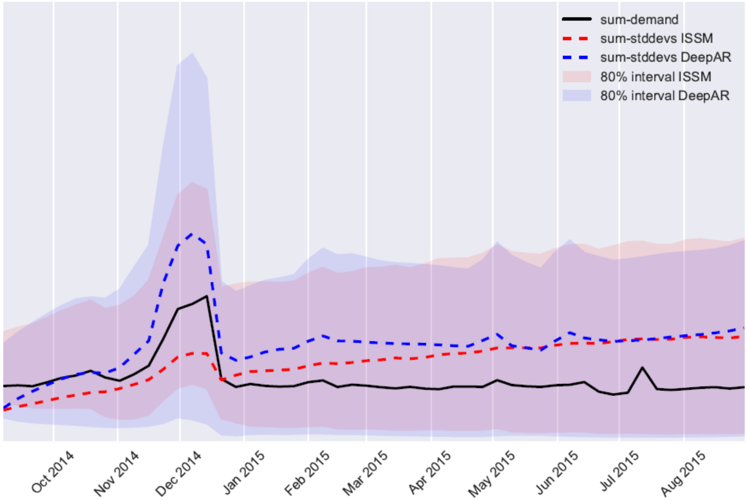
Experiments : comparison

ρ-risk : normalized sum of ρ-quantile losses

	0.5-risk				0.9-risk				average
					parts				
(L, S)	$(0, 1)$	$(2, 1)$	$(0, 8)$	all(8)	$(0, 1)$	$(2, 1)$	$(0, 8)$	all(8)	average
Snyder (baseline)	1.00	1.00	1.00	**1.00**	1.00	1.00	**1.00**	1.00	1.00
rnn-gaussian	1.17	1.49	1.15	1.56	1.02	0.98	1.12	1.04	1.19
rnn-negbin	**0.95**	**0.91**	0.95	**1.00**	1.10	**0.95**	1.06	0.99	0.99
DeepAR	0.98	**0.91**	**0.91**	1.01	**0.90**	**0.95**	**0.96**	**0.94**	**0.94**
					ec-sub				
(L, S)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	average
ISSM (baseline)	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00
rnn-gaussian	1.03	1.19	1.24	0.85	0.91	1.74	2.09	0.67	1.21
rnn-negbin	0.90	0.98	1.11	0.85	1.23	1.67	1.83	0.78	1.17
DeepAR	**0.64**	**0.74**	**0.93**	**0.73**	**0.71**	**0.81**	1.03	**0.57**	**0.77**
					ec				
(L, S)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	$(0, 2)$	$(0, 8)$	$(3, 12)$	all(33)	average
ISSM (baseline)	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00
rnn-gaussian	0.89	0.91	0.94	1.14	0.90	1.15	1.23	**0.90**	1.01
rnn-negbin	0.66	0.71	**0.86**	**0.92**	0.85	1.12	1.33	0.98	0.93
DeepAR	**0.59**	**0.68**	0.99	0.98	**0.76**	**0.88**	**1.00**	0.91	**0.85**

Figure: Accuracy metrics relative to the strongest previously published method

Experiments : comparison



Contents

Introduction

Related Work

Model
- Architecture
- Likelihood model
- Training
- Scale handling
- Prediction

Experiments

Conclusion

Conclusion

$\surd$ Forecasting approaches based on modern deep learning techniques can drastically improve forecast accuracy.

$\surd$ DeepAR

- Effectively learns a global model
- handles widely-varying scales through rescaling
- generates calibrated probabilistic forecasts with high accuracy
- learn complex patterns such as seasonality and uncertainty growth over time from the data

Thank You for attention!