Products identification from text descriptions and forecasting of multiple related time series by RNN + FCNN

Stanislav Novikov
stanislav.novikov@gmail.com
Project on GitHub: https://github.com/Run4rest/novikovsn

Abstract

In some cases of retail and supply management there is no exact identifier of product in shipments. Instead of unambiguous id there are short descriptions with arbitrari abbreviations and typos. However product identification is required to optimize everyday operations. One of these cases is described in [1]. In this research I simulate the problem and develop the idea of the authors to propose:
- solution for extracting maximum information from text descriptions for further forecasting
- NN for forecasting that brings together benefits of FCNN and RNN architectures and might be more efficient than NN presented by [1]

Introduction

This document consists of 3 chapters. Chapter 1 illustrates the case outlined in [1] and gives a brief description of data simulation. The full description could be found in comments of python scripts (see Appendix). Chapter 2 explains the solution of the product identification problem. Chapter 3 presents NN to predict DoS distribution of a given shipment (see [1]). This NN joins the forecasting abilities of FCNN and RNN.

1. Problem formulation and description of data simulation

Problem formulation

The paper [1] focuses on the problem of storage assignment in warehouses. Each shipment has a certain number of pallets. All pallets of a shipment contain one product.
It is necessary to predict the duration of stay (DOS) of pallets of shipment.
The products are mainly food. The record of shipment contains categorical and text info.
To predict DoS, authors assume that for every shipment S which contains multiple pallets, the DoS of a random pallet follows a cumulative distribution $Fs(t)$. To some extent such $Fs(t)$ distribution can be identified using the characteristics of the shipment known at arrival. That means it is possible to estimate the $Fs(t)$ of a new shipment using machine learning.
So for $Fs(t)$, authors exploit that each shipment arriving usually contains $p \gg 1$ units of the good. Let the $i\,th$ unit leave the warehouse at time Ti then we can create empirical cumulative distribution function (CDF):

$$Fs(t) = \frac{1}{p}\sum_{i=1}^{n} 1_{Ti \le t}$$

CDF constructed in this way allows to determine labels as the 5%,⋯95% percentile points. Input data consist of two parts: single-valued and ambiguous. Single-valued are date of arrival, warehouse location, customer type, product group, pallet weight, inbound location, outbound location. The ambiguous part is a text description of the product in pallets. This field describes the content of the pallet, but most of the descriptions are not written in a human readable form, such as ”NY TX TST CLUB PACK”. Acronyms are used liberally due to the length restriction of item descriptions in the computer system. The acronyms do not necessarily represent the same words: ”CKN WG” means ”chicken wing” while ”WG CKN” means ”WG brand chicken”.

Data simulation

The main goal of data simulation is to consider the problem of [1] a little wider. It should be noted that not all details of simulation were used in research explicitly. Big part just tries to imitate reality and can be investigated later.
Simulation is based on normalized demand curve (NDC) and trend/seasonality function (TSF) for a product.

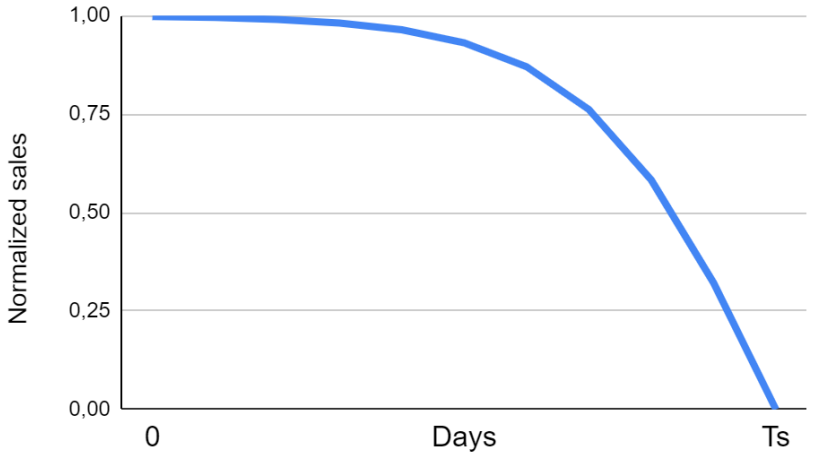
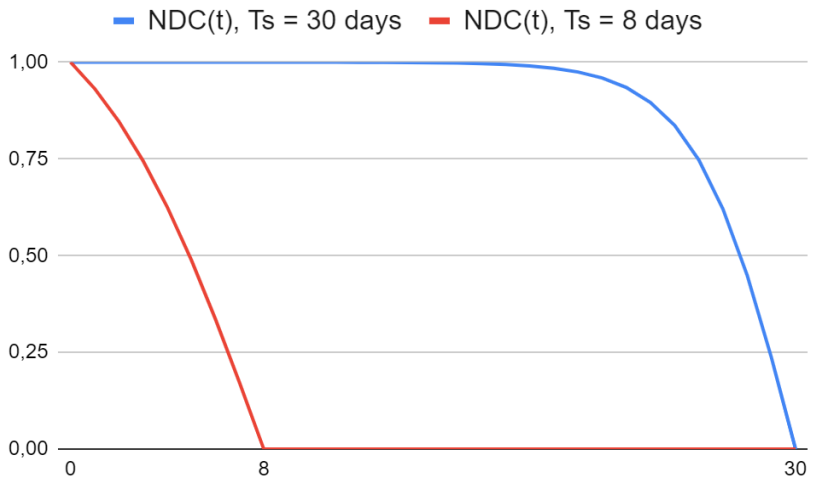


Fig.1 NDC (sales) of pallets

Assume the normalized demand curve of a product in a shipment looks like Fig.1. Demand constantly decreases as product storage life finishes and is equal to zero after expiration period (Ts) .
Let's consider a NDC with $Ts < 10$ and $Ts > 10$ In my opinion the first curve should be more steep. Thereby base demand curve looks like:
$D(t) = Max(-\tanh(k(t - Ts)), 0)$
where, t is time in days, Ts is product expiration period in days, k determines slope of $D(t)$ and can take the following values:

1. $k = 7,25/Ts, Ts > 10\ days$
2. $k = 1,1/Ts, Ts < 10\ days$

Demand of product is equal to NDC multiplied by TSF element wise.



Couple TSFs to test DoS forecasting with quarter and year seasonality are shown on Fig.3

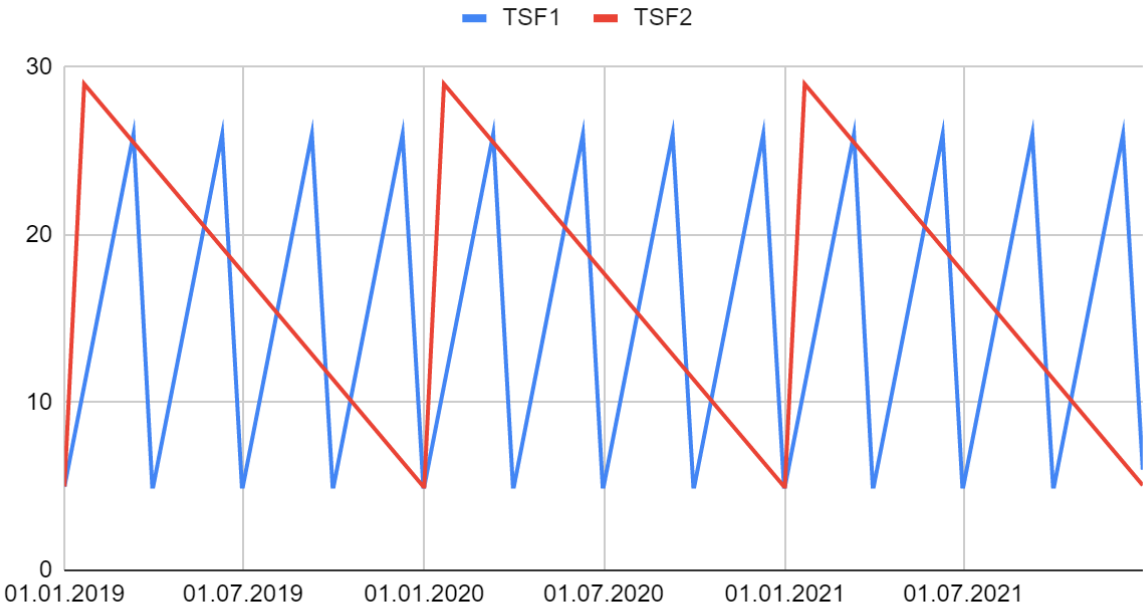


Fig.3 TSFs with quarter and year seasonability

Simulation of text description takes into account the following statement of [1]: "We limit the description to the first five words with zero padding."
Thus the simulated description contains 6 words. 5 words present 5 independent product categories and the 6th word determines supplier. 5 categories are presented only in the description. In addition to the description, the supplier is represented by a separate variable sid. I added suppliers into text in order to consider situations like one above: ”CKN WG” means ”chicken wing” while ”WG CKN” means ”WG brand chicken”.
From this research point of view, the meaning of the categories does not matter. The main idea of the problem is that the demand for a product which, up to a coefficient, is defined as NDC×TSF depends mainly on the textual description.
Nevertheless, I tried to give the categories a 'real' meaning. Five independent categories are:

- product group
- volume of product
- packaging
- content (property describing the content of the product, for example, creamy or fruit ice cream, thick pizza or thin, content of fat)
- nutrient value.

All categories consist of acronyms. See examples below.
Product group: MLK - milk, JGRT - jogurt , ICRM - ice cream, CHEES - cheese
Volume of product: 500ml, 1L, 2L, 500g
Packaging: PBTL - plastic bottle, GBTL - glass bottle, PCNR - plastic canister, PRCNT - paper container,
Content: 3.6%, 5%, 9%, CREM - creamy, FI - fruit ice (for ice cream), TCK - thick, TIN - thin (for pizza)
Nutrient value: LCTFR - lactose free, SGRFR - sugar free, DIET - dietary, ORG - organic
In addition to categories the following non-contradictory cross-references are introduced: supplier × product group, supplier × nutrient value, product group × volume, product group packaging, volume × packaging, product group × content, content × nutrient value
Example of product group × volume of product:

MLK	100ml, 300ml, 500ml, 1L, 2L, 5L
CHEES	100g, 300g, 500g, 1kg

Besides that there are several relationships: client × product, client × client type, client × outbound location, supplier × outbound location.
As result the simulated dataset contains the following features: supplier, client, client type, outbound location and text description containing supplier, product group, volume of product, packaging, content, nutrition value. Let's call this data Shipment Categorical info (SCI).
After merging into one dataset, the number of SCI rows is 1062. Number of unique descriptions in the simulation is 566.
To calculate DoS I use 3 more dataframes. One defines the dependence of Ts on a group of products, packaging, content, nutritional value. Another one sets the dependence of TSF on a product group. And the last one contains coefficients to amplify demand by outbound location (outbound location × coefficient)
DoS calculation algorithm for each SCI line includes several steps:

1. selects the date of the first shipment from the given period
2. calculates NDC by days, multiplies elementwise by TSF and by coefficient for a given outbound location, adds noise and gets a sales chart.
3. summarizes sales of the chart by day. That gives us the maximum potential shipment's volume. And multiplies this quantity by a random coefficient in the range from 0.5 to 0.8
4. The number received is the volume of the first shipment. The algorithm returns with this value to the sales chart and counts the interval for which the received volume is sold. This is the total sale time of the shipment. The DoS distribution we obtain from the sales chart. Duration and sales volume of the first delivery are stored as thresholds.
   Next in the cycle:

5. The date of the shipment is chosen randomly within a few days after the completion of the previous shipment.
6. When calculating the sales chart, the algorithm uses the same formula coefficient × NDC × TSF, but the shipment's volume is now known in advance and is equal to the threshold volume. If the total time for sale of the previous shipment is less than the threshold time by half, then the threshold volume for the current shipment is increased by 2 times. And the threshold time gets the value of the total time for sale of the current shipment.
7. If the demand has fallen so much the shipment's volume cannot be sold, the algorithm divides shipment's volume into 2 parts: maximum possible sales volume calculated according to the chart and unsold part. Shipment's volume becomes equal to the maximum for the sales chart and the unsold part is stored in the variable "nsl". The threshold shipment volume for the next shipment is set to 2 times less than the maximum calculated for the current chart.

The main idea of the algorithm is that shipments follow demand, but with some delay.
To distort the descriptions I introduce 3 variants of the word order and reassign randomly to shipments
Now each of the descriptions is presented in 3 variants. Table 2 contains examples of descriptions with words reordering.

Initial description	3 variants in simulated data
SUP01 CHEES 1kg VACUM 50%	CHEES SUP01 1kg 50% VACUM; SUP01 VACUM 1kg CHEES 50%; SUP01 CHEES 1kg VACUM 50%
SUP03 MLK 2L PBTL 3.6%	SUP03 MLK 2L PBTL 3.6%; SUP03 PBTL 2L MLK 3.6%; MLK SUP03 2L 3.6% PBTL
SUP01 MLK 500ml GBTL 1.5%	MLK SUP01 500ml 1.5% GBTL; SUP01 MLK 500ml GBTL 1.5%; SUP01 GBTL 500ml MLK 1.5%

Table 2. Examples of descriptions with words reordering

As you will see later, the distortions can be anything. It is only important that their number is enough to train the model. It is even possible to augment distortions to improve the quality of learning. However, I have to balance between the hardware capabilities of my laptop and the presentability of the result.
Data were simulated for 6 years. Total number of rows is about 200000. Dataset of simulated data contains columns shown in Table 3

Column	Description
tid	client type of a shipment
cid	client

oid	outbound location
sid	supplier
is	date of a shipment
yy, mm, dd, wd	year of a shipment, month (1..12), day of month (1..31), day of week (1..7)
w0 .. w6	7 words of the text description. 7, not 6, because the supplier's name can consist of 2 words. For example, the name of supplier #5 is 'SUP05 CHEES'. I have introduced such names as additional distortions. These words are reordered as shown in Table 2 and include product group, volume of product, packaging, content, nutrition value.
nsl	unsold part of a shipment
drn	the total sale time of a shipment
qnt	volume of a shipment or sold part of a shipment when nsl more than 0
stop	threshold time
T01 .. T20	the 5%,⋯95% percentile points of CDF
did	id of initial description. It is not used in the product identification problem. In forecasting, this field is present as a result of successful product identification (see Chapter 2).

Table 3. Fields of simulated data

That is not exactly [1] but is quite enough for a model task.

2. The product identification problem

The product identification problem means there is no set of fields that would uniquely identify the product. However such an identifier could help us better forecast DoS. At least we can build a time series of shipments for one product, client and so on and predict DoS of the next shipment using the time series.
To build the identifier the following algorithm is proposed:
1. Build character embedding for descriptions
Here it is necessary to make the following remark. Descriptions mostly consist of abbreviations and contain many typos. The meaning of the descriptions is limited and boils down to listing a small number of features. Therefore, let's consider the language of descriptions as independent. But instead of word embeddings, we build character embeddings on the description texts.
2. Use multihead attention mechanism (MHA) for embedded descriptions and merge MHA output to other features of shipments.
3. Build deep NN for forecasting DoS based on all features.

4. Combine MHA and deep NN in one model and train the model
5. Perform clustering of MHA output
6. Replace descriptions with cluster identifiers.

Scheme of the algorithm is depicted on Fig 4. Table 4 shows examples of grouping descriptions by the proposed algorithm. Table 5 shows a full list of errors of the algorithm version realized in the attached scripts. As you could see about 2% of descriptions were clustered not ideally. I suppose the following reasons for the errors:

- incorrect hyperparameters to build chart embeddings.
  Clusters 4 and 155 contain 2 separate descriptions each where each separate description is presented by 3 variants. I think this problem can be solved by choosing hyperparameters or transformations of the MHA output
- Due to large changes of demand some shipments and, as a consequence, their descriptions were assigned by the algorithm to independent groups/clusters. This is not quite an error for forecasting. Anyway increasing the volume of input data may help.
- Clustering features. These features have to be considered carefully.

Artificially increasing the number of rare errors in descriptions also increases the accuracy of the algorithm. In some cases, errors can be parsed and corrected by hand.
Hyperparameters and transformations of the MHA output prior to inclusion in the input dataset should be the subject of research on a case-by-case basis.

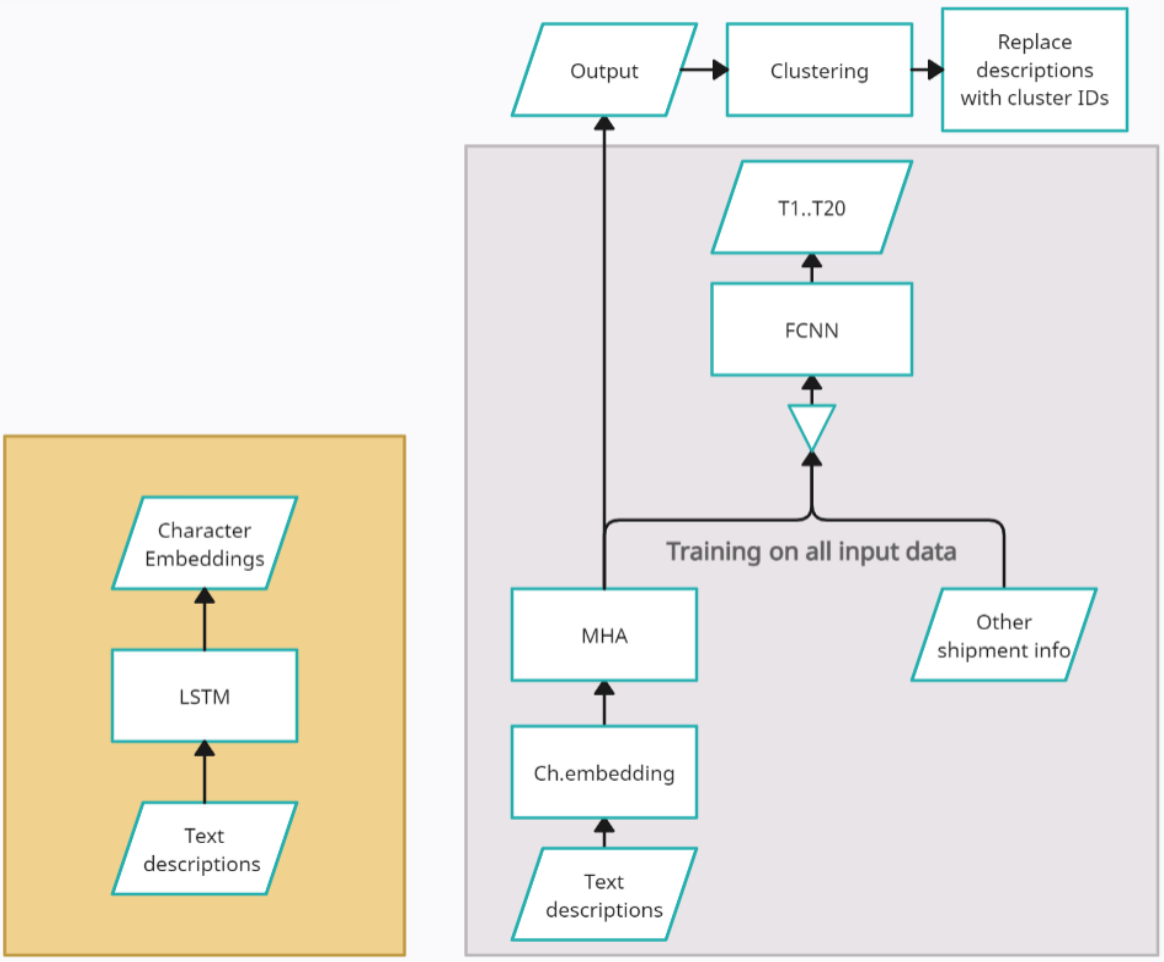


Fig.4. Flowchart diagram of product identification

cluster ID	Clusters of descriptions
1	CHEES SUP01 1kg 50% VACUM , SUP01 VACUM 1kg CHEES 50% , SUP01 CHEES 1kg VACUM 50%
2	SUP03 MLK 2L PBTL 3.6% , SUP03 PBTL 2L MLK 3.6% , MLK SUP03 2L 3.6% PBTL
3	MLK SUP01 500ml 1.5% GBTL , SUP01 MLK 500ml GBTL 1.5% , SUP01 GBTL 500ml MLK 1.5%
5	SUP01 GBTL 1L KFR 5% , KFR SUP01 1L 5% GBTL , SUP01 KFR 1L GBTL 5%
6	SUP01 PCNT 300g CURD FATFR 1.0% , CURD SUP01 300g FATFR 1.0% PCNT , SUP01 CURD FATFR 300g PCNT 1.0%
7	MLK SUP04 1L FATFR 1.0% TPAK , SUP04 MLK FATFR 1L TPAK 1.0% , SUP04 TPAK 1L MLK FATFR 1.0%
8	SUP01 PBTL 300ml KFR 5% , SUP01 KFR 300ml PBTL 5% , KFR SUP01 300ml 5% PBTL
9	SUP01 PCNT 500ml JGRT 9% , JGRT SUP01 500ml 9% PCNT , SUP01 JGRT 500ml PCNT 9%

Table.4. Examples of descriptions grouped by the algorithm into clusters

cluster ID	Clusters of descriptions	Count of descriptions in cluster
4	SUP01 PGLS 300ml SCRM 10% , SCRM SUP01 100ml 30% PGLS , SUP01 PGLS 100ml SCRM 30% , SUP01 SCRM 100ml PGLS 30% , SCRM SUP01 300ml 10% PGLS , SUP01 SCRM 300ml PGLS 10%	6
155	SUP01 PCNR 5L MLK 3.2% , MLK SUP03 2L 1.5% PCNR , SUP01 MLK 5L PCNR 3.2% , MLK SUP01 5L 3.2% PCNR , SUP03 MLK 2L PCNR 1.5% , SUP03 PCNR 2L MLK 1.5%	6
631	SUP01 PCNT 500g CHEES 25% , CHEES SUP01 500g 25% PCNT	2
637	SUP03 MLK FATFR 500ml GBTL 1.5%	1
776	CHEES SUP02 HEAD ORG 60% PCNT , SUP02 PCNT HEAD CHEES ORG 60%	2

Table.5. Errors of the algorithm

3. NN to predict DoS distribution

As soon as we have product identification, standard methods can be used for time series forecasts such as exponential smoothing (ETS) and the autoregressive integrated moving average (ARIMA)

However as stated in [2] Recurrent Neural Networks (RNNs) have become competitive forecasting methods. According to [2] the advantage of RNN is that a single model can learn from many similar time series simultaneously. Guided by this statement let's group RNN and FCNN.

At first we prepare a time series of shipments for each product, client, outbound location, supplier. Every shipment in the series is described by two parts of input data:

- data of the shipment: supplier, client, client type, outbound location, product (cluster ID), day of week (1-7), day of month (1-31), month (1-12)
- the time series till the shipment (TSTS).

Then we feed TSTS to a RNN encoder and add output to the rest of the shipment's features. All features we feed to FCNN. Labels remain the same: the 5%,⋯95% percentile points of CDF.
Architecture of the NN is depicted on Fig 5.

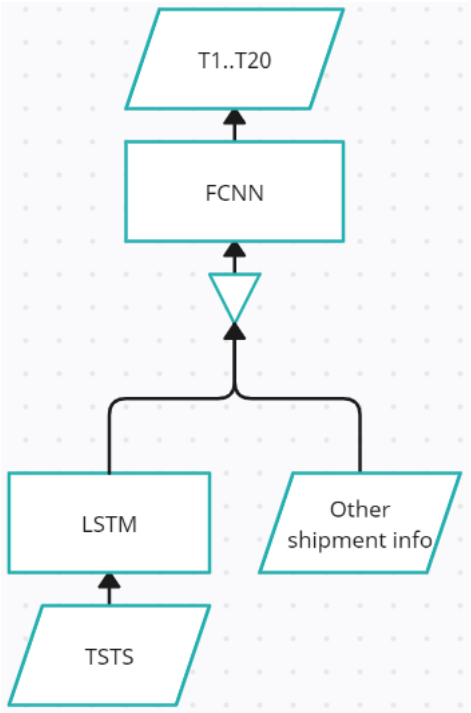


Fig.5. Architecture of NN to forecast DoS after product identification

This architecture can increase forecast accuracy focusing on one or another part of input data in different situations.
To illustrate this statement consider two cases. For both cases a dataset for training will contain the first 5 years of simulated data, a dataset for testing will contain the last 6th year. TSF for the first case (TSF1) and TSF for the second one (TSF2) are depicted in Fig.6.

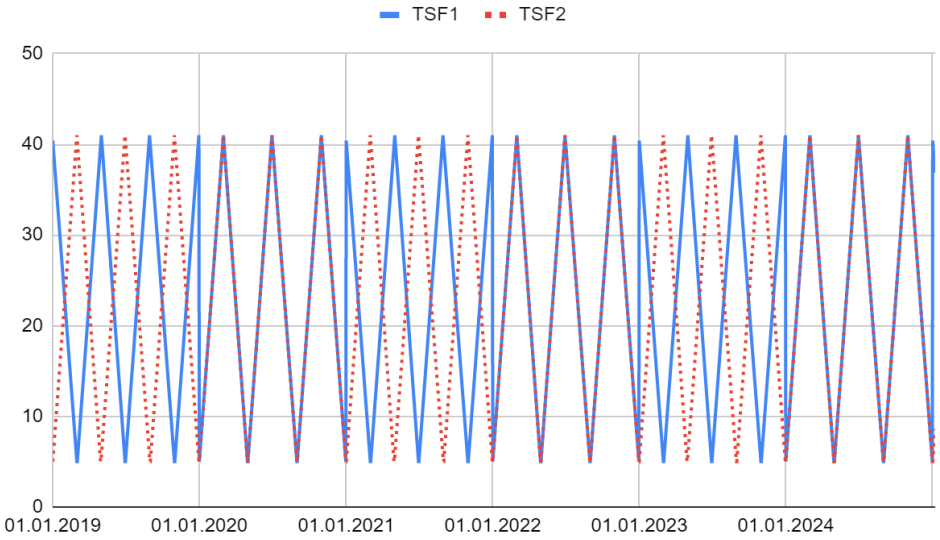


Fig.6 Well (TSF1) and poorly (TSF2) conditioned annual seasonality

TSF1 has clear year seasonality whereas the seasonality of TSF2 lags half a period every odd year.

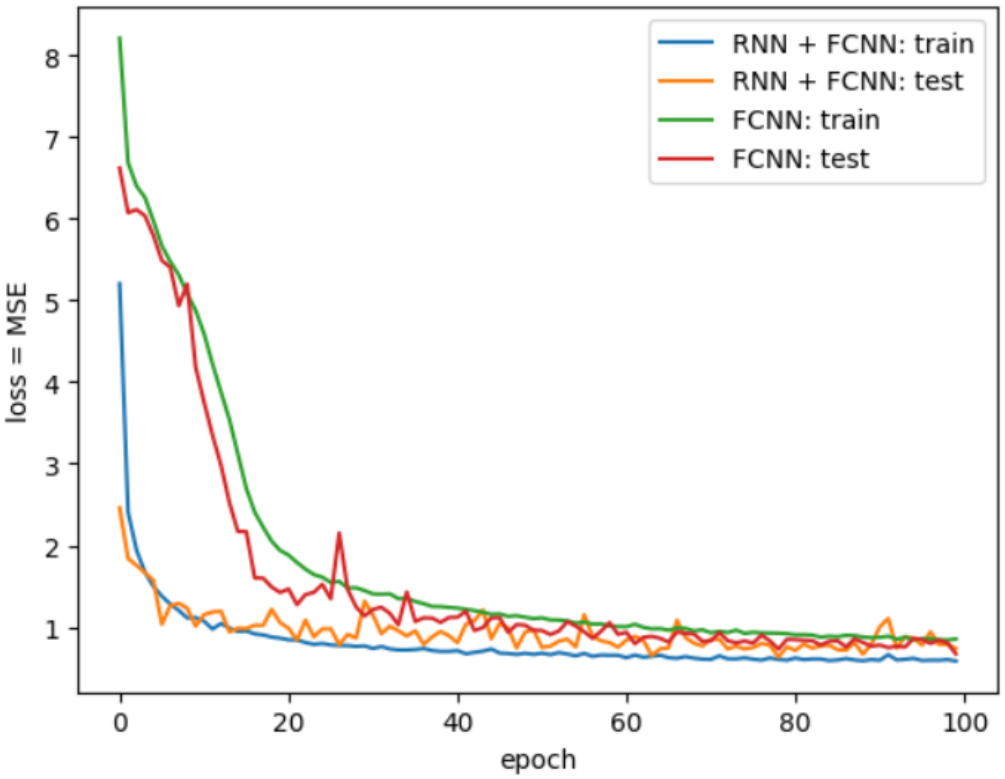


Fig.7 FCNN and RNN + FCNN for TSF1 like seasonality

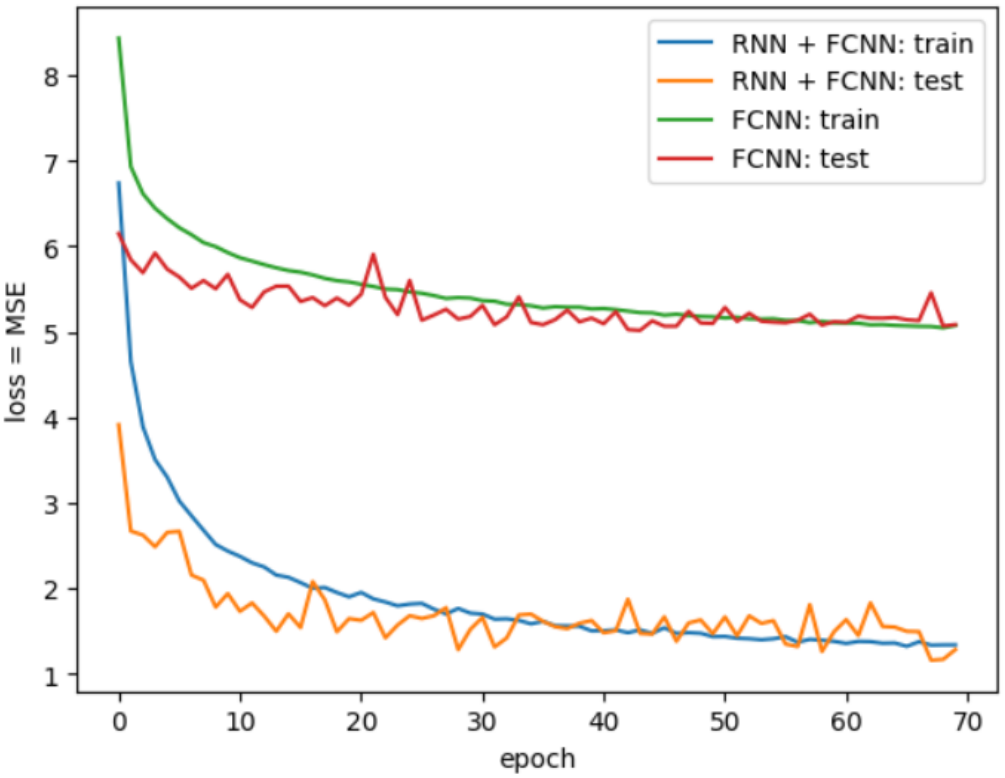


Fig.8 FCNN and RNN + FCNN for TSF2 like seasonality

Let’s build and train two models: RNN + FCNN as shown in Fig.5 and FCNN only. The latest one will not use time series data. The training and validation losses for both cases and both models are plotted in Figures 7 and 8.
For well-conditioned seasonality of TSF1 both models show the same accuracy. Therefore in this case, it can be assumed that the time series are not used. And the main source of accuracy is non-time-ordered shipment data. For TSF2 like seasonality the dispersion of sales on the same day in different years is greater and the accuracy of the forecast is determined by the time series or the RNN part of the model.
In general, FCNN and RNN complement each other. For example, a recurring event from year to year can determine the value of DoS more accurately on a given day/interval than a time series.
I would like to point out that since NN in [1] uses only time-disordered information (authors use RNN for processing of text descriptions only) NN of [1] corresponds to FCNN in this work.

Conclusion

In this work I presented an approach that can significantly compared with [1] increase forecasting DoS accuracy. This approach can be easily generalized for predictive problems based on unstructured textual descriptions.
The next step in the development of the presented solution, I see the unification of the tasks of processing textual descriptions and forecasting in one model. In my opinion, this will require the development of a new module similar to Attention one but for grouping and sorting.

References

1. Michael Lingzhi L, Elliott Wolf, Daniel Wintz (2020): Duration-of-stay storage assignment under uncertainty. Published as a conference paper at ICLR 2020. http://arXiv:1903.05063v3 [cs.LG] 1Feb 2020

2. H. Hewamalage, C. Bergmeir and K. Bandara, Recurrent Neural Networks for Time Series Forecasting: Current status and future directions. International Journal of Forecasting (2020), https://doi.org/10.1016/j.ijforecast.2020.06.008.

3. Xiang T.R. Kong, Miaohui Zhu, Kaida Qin & Pengyu Yan (2021): Demand predictive storage assignment mechanism for flower auction centers, International Journal of Production Research, DOI: 10.1080/00207543.2021.1900617

4. Kasun Bandara, Christoph Bergmeir, Slawek Smyl (2019): Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. https://doi.org/10.1016/j.eswa.2019.112896

5. Character-level text generation with LSTM: https://keras.io/examples/generative/lstm_character_level_text_generation/

Appendix

Scripts are used in this work presented in Table A.

Script	**Description**
data simulation eng v1.1.ipynb	This script generates data, simulating the history of shipments to a warehouse.
model p1 v1.1.ipynb	This script performs product identification. It builds/trains MHA + FCNN model and performs clustering of MHA output. See Fig.4
model p2 v1.3.ipynb	This one generates training and test data sets for the FCNN + RNN predictive model, creates and trains the model. Writes the learning history to the file 'history_p2.txt'
model p3 v1.3	This script generates training and test data sets for the predictive FCNN model, creates and trains a predictive model. The learning history is written to the file 'history_p3.txt'

Table A Scripts of this work.