Bayesian Neural Models for Time-Series Prediction of CS28 Compressive Strength in Cement Manufacturing

Thiago I. A. Lira[1], **Marcelo Finger**[1]

[1]Department of Computer Science
Institute of Mathematics and Statistics
University of São Paulo (IME-USP) – São Paulo, SP – Brazil

{thlira,mfinger}@ime.usp.br

Abstract. *Given the increasing importance of the prediction of cement compressive strength for more efficient use of resources by the industry, recent literature has been experimenting with statistical models to aid the industrial process. This work studies the application of Bayesian Deep Learning (DL) techniques to achieve robust and accurate predictions of compressive strength. The positive results obtained pave the way for similar models to be integrated on day-to-day decision making on the factory floor.*

Resumo. *Dada a crescente importância da predição da resistência compressiva do cimento para o uso mais eficiente dos recursos da indústria, trabalhos recentes tem experimentado com modelos estatísticos para auxiliar o processo industrial. Esse trabalho estuda a aplicação de Aprendizagem Profunda Bayesiana para obtensão de predições robustas de resistência compressiva. Nosso trabalho é um caminho para que modelos similares possam no futuro serem integrados ao processo de tomada de decisão no chão de fábrica.*

1. Introduction

Concrete is among the most manufactured products in the world, both in volume and in value. Therefore the prediction of cement compressive strength (CCS) has been recognized of great importance to the civil construction industry, given that this capacity is one of the factors to assess the quality of cement.

The 28-day compressive strength (CS28) is chosen as the measurement used as a regulatory parameter in construction codes all over the world, and the variance of these measurements indicates the stability of the industrial process [Soroka and Stern 1976, Çolak 2006, Ramezanianpour and Hooton 2014].

It is important for industry to have a system that, detecting falling values on that parameter, can point to anomalies on the process (e.g. concentration of reagents) probably causing it. The problem can them be tackled with regression, where the target is the CS28 measurements and the inputs are chemical compositions and other quantities measured on a daily basis on the factory.

The goal of this paper is to propose a new approach for CCS prediction, implementing Bayesian Deep Learning techniques to leverage the scalability of Deep Learning with the uncertainty estimation of Bayesian Statistics. We will consider all cement data as time series (i.e. indexed by time), with the assumption that the past of the measurements

have some information concerning the future of the process. We extend previous laboratorial approaches [Soroka and Stern 1976, Çolak 2006, Ramezanianpour and Hooton 2014] to account for the dynamical nature of the industrial process, providing an accurate and safe analysis that can be made on a daily basis on the industrial floor.

We will then combine these new techniques to leverage our amount of data, improving upon past work on CCS prediction. We shall employ the same method of training multiple models on different subsets of features as proposed on [Tsamatsoulis 2015]. The main contribution of this paper is a comparison between this past work using dynamical coupled linear models [Tsamatsoulis 2015] and 3 sequence-to-sequence Bayesian DL models being applied on this novel domain for the first time. Our method has outperformed a previous baseline in the prediction of the RC28 metric.

2. Review of the Literature

There is an increasing amount of work done with supervised learning for the prediction of CS28 and related metrics. Past work has focused on multiple regression [García-Casillas et al. 2007, Tsamatsoulis 2014], and more recent work employs Artificial Neural Networks [Kumar and Naranje 2019, Kumar et al. 2020, Zhang et al. 2012]. As the amount of available data grows, so does the complexity of models that one is able to use to work with the data. If past works focused on Ordinary Least Squares (OLS) analysis with dozens of data points, and more recent work used Artificial Neural Networks (ANNs) with hundreds of data points, we are now able to go further with tens of thousands of data points and utilize Deep Learning methods.

Another characteristic of some past work is the focus on laboratorial analysis, with data that is generated many days after that particular batch of cement was made [García-Casillas et al. 2007].

The first work to make this distinction between dynamic and static models to predict the CCS was [Tsamatsoulis 2014]. The authors propose a dynamic moving window linear regression that recalculates it's regression parameters every time a new CS28 measurement is ready. This dynamical model is also able to provide standard deviations for the regression parameters across time.

For many decades the state-of-the-art on forecasting has been the Autoregressive integrated moving average (ARIMA) class of models, and Box-Jenkins procedures [Asteriou and Hall 2016], but these models have their limitations. They are unable to model non-linear relationships on the data [Green et al. 2011], and many real world processes happen to be non-linear. These models also have trouble using exogenous variables to aid the time series forecasting [Laptev et al. 2017].

Recently, many time series problems on different domains had their state of the art results improved by new Deep Learning methods such as Recurrent Neural Network models [Marino et al. 2016, Berriel et al. 2017], these deep learning models require no *feature engineering* (i.e. specific knowledge about the data) and work very well even with gigabytes of data [Asteriou and Hall 2016]. They are more data-driven than classical methods and assume very little about the data, meaning that little manual tweaking on the model parameters is necessary.

Classic time series models fail to scale well to huge amounts of data, although

they are able to provide good uncertainty estimates [Flunkert et al. 2017]. Past work [Robles et al. 2008, Khashei and Bijari 2010] has managed to combine ARIMA models Artificial Neural Networks (ANN), with better results than just either of these models.

3. The problem of prediction of Compressive Strength

Our target of interest will be CS28 measurements (y) and our independent variables will be multiple chemical and physical properties also measured on the factory floor. The training data for the models is composed of the pairs $(\{\mathbf{x}_{t_0}, y_{t_o}\}, \{\mathbf{x}_{t_1}, y_{t_1}\}, \ldots, \{\mathbf{x}_T, y_T\})$ on the time span $[t_o, T]$, where x is a vector of inputs. Let F be a finite time horizon, such that $F > T$, the models should learn to output a probability distribution of the form:

$$p(y_{T:F}|y_{t_o:T}, \mathbf{x}_{t_0:T}).$$

representing the modeling of the distribution of the next $(F - T)$ y values, given some past horizon of x and y values.

3.1. Performance Metrics

We use two performance metrics to validate the performance of our models. Root Mean Square Error (RMSE), which is common for regression problems [Goodfellow et al. 2016] and Coverage, which is extensively used on time series forecasting literature [Maddix et al. 2018, Laptev et al. 2017, Flunkert et al. 2017]. Let y and $\hat{y}$ be vectors of, respectively, the true targets and the predictions of the models.

The RMSE is defined as:

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

For one vector of true targets y and one vector of predictions $\hat{y}$, with it's associated estimated μ_i and σ_i for each entry, we can calculate the coverage that the predictions have on the true values of y. For a given quantile ρ (e.g. 90% confidence interval, or $\rho = 0.9$), we can calculate the confidence interval for the true values of y_i with upper and lower bounds given by $\mu_i \pm z_i^\rho \frac{\sigma_i}{\sqrt{n}}$, where z^ρ is given by a z-table. The coverage of the predictions for the quantile ρ is defined as:

$$Coverage(y, \hat{y}, \rho) := \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left[\hat{y}_i + z_i^\rho \frac{\sigma_i}{\sqrt{n}} > y_i > \hat{y}_i - z_i^\rho \frac{\sigma_i}{\sqrt{n}}\right].$$

For example, a coverage of 0.6 for the 0.9 quantile would indicate that our uncertainty estimations captured only 60% of the true values for each confidence interval that for every distribution $\hat{y}_i \sim \mathcal{N}(\mu_i, \sigma_i)$ would theoretically cover 90% of the observations.

3.2. Dynamic Linear Regression

To compare with Bayesian DL, we will build upon the work done by [García-Casillas et al. 2007], whose authors propose a Coupled Linear Model, which is trained on 3 different sets of features for the same data. In analogy, we will have one

model that uses CS1 to predict the CS28 and one that uses CS1, CS3 and CS7. The main idea is that the former model predicts CS after 28 days based on CS in the first day, so the latter is "corrected" with the outputs from days 3 and 7 (we have to wait for the CS7 measurements to be ready, which means that this model with more features can only be employed at least 7 days after the cement is ready).

These linear models are then re-trained after a fixed number of days with new data.

Exponential Smoothing. The authors further propose to use the models trained with features that take longer to measure to **correct** models with quicker to measure features (e.g. the same day the cement is ready). Following this idea we will correct the CS1 models with the CS7 models, both for the linear regression models and the DL models. The correction term begins with calculating the absolute error vector (we will show which time steps are used shortly) of the CS7 model error:

$$diff_i = \hat{CS28}_i^{reglin\_7} - CS28_i.$$

We shall then calculate the moving average (filtering) for the difference vector, with decay parameter α. This calculation yields the vector $diff\_ew$:

$$\begin{aligned} diff\_ew_0 &= diff_0 \\ diff\_ew_t &= (1-\alpha)diff_{t-1} + \alpha diff_t \ . \end{aligned}$$

The predictions from the CS1 models are then corrected with the filtered vector diff_ew. The time intervals used for such corrections are $[T-1, T-1+t_f]$ for the outputs of the CS1 model and $[T-29, T-29+t_f]$ for $diff\_ew$.

$$\hat{CS28}_t^{reglin\_ew} = \hat{CS28}_t^{reglin\_1} + k * diff\_ew_t.$$

4. Our Approach

Our approach consists of training models for each set of variables as explained on Section 3.2, but we will use Deep Learning models instead of OLS. We experimented with the 3 DL models with state of the art results for time series forecasting. We used the implementation of the DeepAR[Flunkert et al. 2017] and DeepFactors[Flunkert et al. 2017] models from the GluonTS [Alexandrov et al. 2019] library. The Encoder Decoder Forecaster model [Laptev et al. 2017] was implemented using primitives provided by the PyTorch library. We fitted 3 variants for each of those models for the CS prediction task, as comparison to the previous best method of using a dynamic OLS [Tsamatsoulis 2014]. As previously stated, we will fit 3 variants for each model, following the idea in [Tsamatsoulis 2014]. Each variant will be trained with a different set of features, simulating models that could be used in "real time". If we are currently on day t the "1" models will be able to use all data up to the last day, the "3" models will be able to use data up to $t-4$ and "7" models will be able to use data until $t-8$. As explained on the previous Section, this is due to the idiosyncrasies of the data, in which some features take longer to be available than others for the cement data.

Data Source. Our data was provided to us from the company Intercement. The data was collected for control purposes from the Cajati factory, on a daily basis, for a period of 11 years. There are multiple spreadsheets with data collected at different times on the manufacturing process, but we will use only the data corresponding to chemical and physical analysis of cement samples just before shipping i.e. just before being bagged. Table 1 gives an overview of our input variables from this data.

Variables (unit)						
Chemical Composition (%)	AL_20_3	SIO_2	MGO	RICARB	P_2O_5	F_2O_3
Water (%)	AGP					
Time until beginning and end of material hardening (s)	IP	FP				
Blaine Fineness (cm^2/g)	SBL					
Compressive Strength (kPA)	RC1	RC3	RC7	RC28		

Table 1. Overview of input variables contained within the data provided to us by the cement factory.

The first two rows of features are percentage of mass in the final product, e.g. the feature AL_20_3 is the percentage of Aluminium oxide in the cement and the feature AGP is the percentage of water. SBL is the Blaine Fineness of the cement, which is the surface area per unit of mass of the cement. RCX are Compressive Strengths, which is the capacity of a material or structure to withstand loads tending to reduce size [Agency 2009], measured in kilopascals (kPa).

Training and Inference. All DL models are based on Long Short-Term Memory Networks (LSTMs) [Hochreiter and Schmidhuber 1997], and have two distinct regimes in respect to their use: training and inference. During training we create multiple data points by separating windows of pairs of inputs and outputs. With the size of this window being itself a hyper-parameter w, we shall have multiple windows for many values of t from the training data of the form:

$$((CS28_t, \mathbf{x}_t), (CS28_{t+1}, \mathbf{x}_{t+1}), \ldots, (CS28_{t+w}, \mathbf{x}_{t+w}))$$

The task for the models during training is to consume series of inputs $\mathbf{x}_t$ and output some prediction or a series of predictions, which will then be compared with the real CS28 for the corresponding timestep. Since the models we employ on this work are probabilistic, the output is in fact sampled from a probability distribution of the target CS28 output. We shall then illustrate these two regimes on the DeepAR model [Flunkert et al. 2017] on Figures 1 and 2

The DeepAR model works by deterministically calculating parameters from the output's probability distribution (for the Gaussian case μ_t and σ_t) for each timestep from it's internal state h_t. During inference the models shall learn to generate new outputs in a auto-regressive fashion, i.e., by being fed the last predicted output while generating a sequence. The outputs are **sampled** from the probability distribution given by the model for each timestep. The main benefit of using Bayesian models comes from the fact that we are then able to estimate the posterior distribution of each output given by the model. Both [Maddix et al. 2018] and [Flunkert et al. 2017] use techniques to propagate

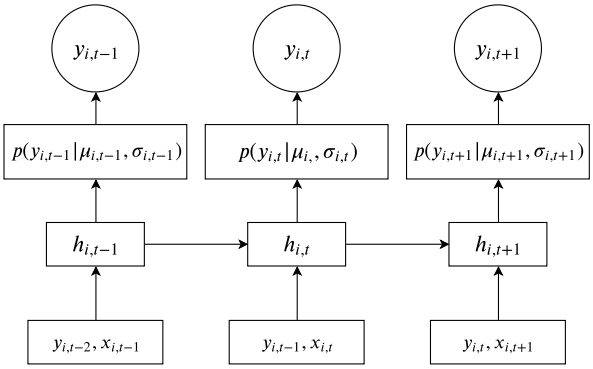

Figure 1. Training regime for the DeepAR sequence-to-sequence model.

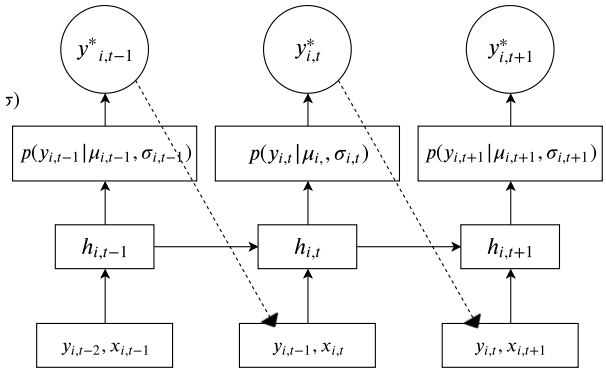

Figure 2. Inference regime for the DeepAR sequence-to-sequence model.

the uncertainty from the models to the outputs. We can then estimate quantiles for each individual output by simply sampling from the probability distribution calculated on each timestep.

4.1. Our Model

We propose a model similar in operation to the linear models proposed in [Tsamatsoulis 2014], but with a DL architecture. Figure 3 illustrates our variant that uses both CS1 and CS3, equivalent to the model shown in Figure 4. All our models are then corrected with exponential smoothing to stabilize the predictions as is done in [Tsamatsoulis 2014].

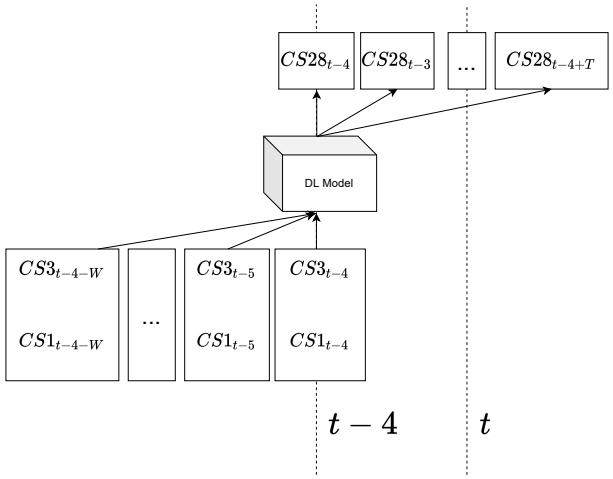

Figure 3. The Deep Learning models, on inference time. The model shall consume a window of data of size W and generate a new window of T CS28 results.

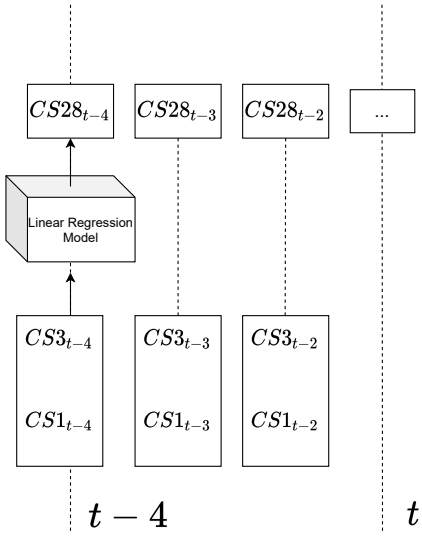

Figure 4. The regression models, on inference time

5. Results and Discussion

Tables 2 and 3 compare the probabilistic coverage yielded by each version of each Bayesian Deep Learning model.

	Deep Factors			Uber Encoder-Decoder			DeepAR		
	1-Model	3-Model	7-Model	1-Model	3-Model	7-Model	1-Model	3-Model	7-Model
0.9 Coverage 1 day	0.89	1	1	0.52	0.57	0.54	0.81	0.75	0.76
0.9 Coverage 7 days	0.89	1	1	0.71	0.71	0.71	0.81	0.75	0.76

Table 2. Coverages of the probabilistic predictions for the 90% quantiles.

	Deep Factors			Uber Encoder-Decoder			DeepAR		
	1-Model	3-Model	7-Model	1-Model	3-Model	7-Model	1-Model	3-Model	7-Model
0.5 Coverage 1 day	0.5	1	0.94	0.24	0.27	0.26	0.41	0.41	0.32
0.5 Coverage 7 days	0.5	1	0.94	0.28	0.28	0.28	0.41	0.41	0.32

Table 3. Coverages of the probabilistic predictions for the 50% quantiles.

Table 4 reports our RMSE results yielded by the DeepAR model, our best performing model (as error is being measured, smaller is better), compared with the state of the art dynamical regression model. It makes clear that our models outperforms linear regression OLS based solutions.

	reglin_1	DeepAR_1	reglin_3	DeepAR_3	reglin_7	DeepAR_7	reglin_ew	DeepAR_ew
RMSE 1 day	1.66	1.68	2.12	1.86	2.09	1.82	2.12	1.16
RMSE 7 days	2.19	1.75	2.02	1.68	1.63	1.42	1.42	1.12

Table 4. Comparing our model with the state-of-the art dynamical linear regression. The values are painted with red where our models underperforms in relation to the OLS model, and blue where it outperforms such models

The experiments serve as a basis to show that DeepAR CS1 model with the correction term outperforms every other model we have tested for the problem of predicting the compressive strength for our data. Also, the DeepFactors based models have the best probabilistic coverage for the 50% and 90% quantiles, though the DeepAR models consistently achieve 0.7 coverage for the 90% quantile. We attribute the success of the DeepAR model to the fact that the model works well with smaller datasets, as is our case in this work. Other models such as the DeepFactors model require many more aligned time series to train, as the attention mechanism needs more data to converge.

Future work involves deep learning architectures with larger context-sensitivity, such as Transformers (which also extensively benefits from using an attention mechanism), however that would require larger quantities of production data from several cement plants.

Acknowledgements

This work was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. Thiago Lira was partly supported by CAPES. Marcelo Finger was partly supported by the São Paulo Research Foundation (FAPESP) (grants 2020/06443-5, 2014/12236-1); and the National Council for Scientific and Technological Development (CNPq) (grant PQ 303609/2018-4).

References

Agency, I. E. (2009). *Cement Technology Roadmap: Carbon Emissions Reductions up to 2050*.

Alexandrov, A., Benidis, K., Bohlke-Schneider, M., Flunkert, V., Gasthaus, J., Januschowski, T., Maddix, D. C., Rangapuram, S., Salinas, D., Schulz, J., Stella, L., Türkmen, A. C., and Wang, Y. (2019). Gluonts: Probabilistic time series models in python.

Asteriou, D. and Hall, S. (2016). *ARIMA Models and the Box-Jenkins Methodology*, pages 275–296.

Berriel, R., Teixeira Lopes, A., Rodrigues, A., Varejao, F., and Oliveira-Santos, T. (2017). Monthly energy consumption forecast: A deep learning approach. pages 4283–4290.

Flunkert, V., Salinas, D., and Gasthaus, J. (2017). Deepar: Probabilistic forecasting with autoregressive recurrent networks. *arXiv preprint arXiv:1704.04110*.

García-Casillas, P. E., Martinez, C. A., Montes, H. C., and García-Luna, A. (2007). Prediction of portland cement strength using statistical methods. *Materials and Manufacturing Processes*, 22(3):333–336.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Green, K., Graefe, A., and Armstrong, J. (2011). *Forecasting Principles*, pages 527–534.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. 9:1735–80.

Khashei, M. and Bijari, M. (2010). An artificial neural network (p,d,q) model for time-series forecasting. *Expert Systems with Applications*, 37(1):479 – 489.

Kumar, N. and Naranje, V. (2019). Prediction of cement strength using machine learning approach. In *2019 Amity International Conference on Artificial Intelligence (AICAI)*, pages 239–243.

Kumar, N., Naranje, V., and Salunkhe, S. (2020). Cement strength prediction using cloud-based machine learning techniques. *Journal of Structural Integrity and Maintenance*, 5(4):244–251.

Laptev, N., Yosinski, J., Li, L. E., and Smyl, S. (2017). Time-series extreme event forecasting with neural networks at uber.

Maddix, D. C., Wang, Y., and Smola, A. (2018). Deep factors with gaussian processes for forecasting. *arXiv preprint arXiv:1812.00098*.

Marino, D. L., Amarasinghe, K., and Manic, M. (2016). Building energy load forecasting using deep neural networks. *CoRR*, abs/1610.09460.

Ramezanianpour, A. M. and Hooton, R. D. (2014). A study on hydration, compressive strength, and porosity of portland-limestone cement mixes containing scms. *Cement and Concrete Composites*, 51:1–13.

Robles, L. A., Ortega, J. C., Fu, J. S., Reed, G. D., Chow, J. C., Watson, J. G., and Moncada-Herrera, J. A. (2008). A hybrid arima and artificial neural networks model to forecast particulate matter in urban areas: The case of temuco, chile. *Atmospheric Environment*, 42(35):8331 – 8340.

Soroka, I. and Stern, N. (1976). Calcareous fillers and the compressive strength of portland cement. *Cement and Concrete Research*, 6(3):367–376.

Tsamatsoulis, D. (2014). Application of the static and dynamic models in predicting the future strength of portland cements.

Tsamatsoulis, D. (2015). Optimizing the cement compressive strength prediction by applying coupled linear models.

Zhang, Q., Yang, B., Wang, L., and Zhu, F. (2012). Predicting cement compressive strength using double-layer multi-expression programming. In *2012 Fourth International Conference on Computational and Information Sciences*, pages 94–97.

Çolak, A. (2006). A new model for the estimation of compressive strength of portland cement concrete. *Cement and Concrete Research*, 36(7):1409–1413.