Goldilocks: An Active Sampling Bandit That's Just Right for Multi-Task Forecasting

Vincent Quenneville-Bélair
quennv@amazon.com
Amazon
USA

Abstract

Neural networks have lead to improvements in demand forecast accuracy for supply chain and retailers. These neural networks have been designed and trained on data representing their particular use cases. We investigate the zero-shot performance of those deep learning models on retail dataset outside of their original use case. As such, we focus on the hypothesis that this zero-shot performance of deep learning models is linked to how we train a model and balance multiple tasks. To address this, we introduce a new active sampling bandit called Goldilocks that samples across multiple tasks, here corresponding to difference velocity groups, based on learnable samples that are not too hard, not too easy, but just right. For comparison, we also offer a novel Dynamic Importance Sampling (DIS), an extension of Static Importance Sampling (SIS) based on demand, used to train neural networks [7, 11]. A temperature hyperparameter in Goldilocks controls the algorithm's preference for harder problems, extending the idea behind DIS. We show out-of-sample convergence results on a public retail dataset called M5 to evaluate the zero-shot performance of the sampling strategies.

CCS Concepts

• Applied computing → Forecasting; • Computing methodologies → Planning under uncertainty.

Keywords

Artificial Intelligence, Learning

ACM Reference Format:
Vincent Quenneville-Bélair. . Goldilocks: An Active Sampling Bandit That's Just Right for Multi-Task Forecasting. In Proceedings of Workshop on AI for Supply Chain: Today and Future (AI4SupplyChain). ACM, New York, NY, USA, 4 pages.

To support downstream inventory management decisions, the supply chain consumes distributional forecasts of demand. The quantile of interest to the downstream customer depends on the application in question. For example, when we consider purchasing a product for the retail market, we are typically interested in forecast values at or above the 50th quantile of the distribution. In contrast, when we are interested in deciding whether to remove or not a product that is not selling as expected from the inventory, forecasts values at or below the 10th quantile are of greater interest.

Demand forecasts are thus currently generated via quantile regression wherein two quantiles (the 50th and 90th) are predicted by a deep learning model. Neural networks have lead to improvements in demand forecast accuracy; starting with MQ-RNN and MQ-CNN in Wen et al. [10] and, more recently, an implementation of a Transformer architecture [9] in MQ-Transformer [3, 4] and in SPADE [11]. These models generate forecasts for a product based on its demand history, product attributes and known future information (e.g. promotions, planned sales, holidays). These neural networks have been designed and trained by sampling by product velocities – the total demand per year – on data representing their particular use cases.

We are interested in understanding the forecasting performance on retail tasks outside of where the model was trained. As a proxy for this zero-shot performance, we evaluate the models on the well-known public retail dataset called M5 [5, 6]. This public dataset represents sales for a large retailer on which the models were not trained on, for a time period outside of the training window used to trained the neural networks in this paper.

We focus on the hypothesis that this zero-shot performance of deep learning models is linked to how we train a model and balance multiple tasks. As such, we focus on curriculum learning – adapting the curriculum used for training by introducing a new active sampling bandit called Goldilocks that samples across multiple tasks, here corresponding to different velocity groups of products, based on samples that are not too hard, not too easy, but just right to learn. This extends the learnability idea from Rutherford et al. [8] wherein it was shown that an intuitive notion of learnability allowed an agent outperformed existing methods in several binary-outcome environments.

For comparison, we also offer a novel Dynamic Importance Sampling (DIS), an extension of Static Importance Sampling (SIS) based on demand used to train neural networks similar to [7, 11]. The extension is dynamic in the sense that the sampling starts proportional to total demand in one year of training, as is done with importance sampling, but then the sampling weights are updated by the latest weighted quantile error. This extension is natural to the business problem of forecasting for supply chain as it helps ensure that the overall quantile loss is low across velocities. A temperature hyperparameter in Goldilocks controls the algorithm's preference for harder problems, extending the idea behind DIS.

In this paper, after introducing the metrics and algorithm, we show convergence results of convergence on a large online retailer and and the public retail dataset M5 to evaluate the zero-shot out-of-sample performance of the models.

AI4SupplyChain, Toronto, Canada

1 Metric: WCRPS

Beyond point estimation, the supply chain management decisions to be made requires a measure of uncertainty that distributional forecasts provide. The quantile of interest depends on the application in question. For example, when we are considering purchasing an product for the retail market, we are typically interested in forecast values at or above the 50th quantile of the distribution. In contrast, when we are interested in marking down the price of a product that is not selling as expected, forecast values at or below the 10th quantile are of greater interest. We thus generate a demand forecast via quantile regression wherein two quantiles (the 50th and 90th) are predicted by a deep learning model.

For a given forecast f, demand d, product a, horizon h, percentile p, for forecast start date t, the Quantile Loss is

$$\begin{aligned}&\mathrm{QL}_p(f_{a,t,h,p}, d_{a,t,h})\\ &\qquad := p\cdot(d_{a,h} - f_{a,h,p})^+ + (1-p)\cdot(d_{a,h} - f_{a,h,p})^-\end{aligned}$$

with superscripts + and - denote the positive and negative parts. The demand-Weighted Quantile Loss (WQL) is

$$\mathrm{WQL}_p(f,d) := \frac{\sum_{a,t,h}\mathrm{QL}_p(f,d)}{\sum_{a,h} d_{a,h}}$$

where we sum over all products and all horizons of interests. WQL is non-negative, and lower is better. The zero forecast gets $\mathrm{WQL}_{90}(0,d) = 0.9$ whereas the perfect forecast gets $\mathrm{WQL}_{50}(d,d) = \mathrm{WQL}_{90}(d,d) = 0$.

Continuous Ranked Probability Score (CRPS) is the integral of the quantile function (inverse CDF, F^{-1}) over the percentiles,

$$\mathrm{CRPS}(F,y) = \int_0^1 \mathrm{QL}_p(F^{-1}(p), y)\,dp$$

where F is the CDF, and y is the target value, see Berrisch and Ziel [2]. For easier comparison with WQL, we omitted a factor of 2 in front of the integral. For a neural network f_{θ,p_i} quantiles at percentiles p_i, we can approximate CRPS using Riemann sum as

$$\mathrm{CRPS} \approx \sum_{p_i} \mathrm{QL}_{p_i}(f_{\theta,p_i}, y)\,\Delta_i$$

where the weights Δ_i chosen constant distances between percentiles and sum to 1. Similarly, we define the demand-Weighted CRPS (WCRPS) is

$$\begin{aligned}\mathrm{WCRPS}(F,y) &= \int_0^1 \mathrm{WQL}_p(F^{-1}(p), y)\,dp\\ &\approx \sum_{p_i} \mathrm{WQL}_{p_i}(f_{\theta,p_i}, y)\,\Delta_i\end{aligned}$$

which is just CRPS normalized by total demand as for WQL. Unlike Quenneville-Belair et al. [7], where the quantiles where weighted by empirical evidence collected from applications, for simplicity here, we define the WCRPS for P_{50}, and P_{90} with a simple symmetric midpoint weighted average, and provides a simple overall comparison point:

$$\mathrm{WCRPS}_{50,90} = 0.7\cdot\mathrm{WQL}_{50} + 0.3\cdot\mathrm{WQL}_{90}$$

which, asymptotically with the number of percentiles, approximate the integral, see Appendix A for a similar discussion for the mean. The zero forecast gets $\mathrm{WCRPS}_{50,90}(0,d) = 0.7\cdot 0.5 + 0.3\cdot 0.9 = 0.62$ whereas the perfect forecast gets $\mathrm{WQL}_{50}(d,d) = \mathrm{WQL}_{90}(d,d) = 0$.

2 Algorithm: Goldilocks

In our application for supply chain, we care about each unit of demand equally (as a lost sale or a product stored), and the quantile loss accounts for that. However, given the number of products being in the millions, and the number of training steps being the thousands, we do not expect to sample the same product multiple times. As such, we describe a product segmentation strategy, where the segments correspond to equally important tasks to forecast.

We assume that ergodicity holds – where the time average of one individual (or product) is equivalent to the ensemble average across many individuals (or products). This allows us to focus on sampling across products with their full history as is done with forking sequences [3, 4, 7, 11], instead of sampling products and time windows.

Here, the products are thus assigned to segments in a way that gives them equal total demand. More precisely, we order products by total demand for one year of training, and group them into a pre-determined number of segments so that each segment has the same total demand. The products with zero demand can be left in a segment at the end, but dropped in this paper.

Since segments are now task of equal importance, we can now sample across them. The idea is then to sample the segments according to probabilities $p(s)$ depending on the WCRPS w_s of the products in each segment s. For DIS, the probability of sampling each segment is simply taken as WCRPS for recent products in that segment,

$$p(s) = w_s \tag{1}$$

and this means sampling products that are difficult to learn. In contrast, for Goldilocks, we extend the learnability idea from Rutherford et al. [8] to our case with WCRPS w_s

$$p(s) = (\gamma w_{\max} - w_s)(w_s - \gamma^{-1} w_{\min}) \tag{2}$$

where the temperature γ controls the emphasis on lower or higher loss, and indicates our preference for harder problem, extending the idea behind DIS. More precisely, $\gamma = 1$ maxes out at the midpoint between the minimum and maximum loss, giving us products that are not too hard, not too easy, but just right to predict. In contrast, $\gamma = 2$ maxes out near the maximum, similarly to DIS, see Appendix B for more details. We detail Goldilocks and DIS in Algorithm 1.

3 Experimental Setup and Results

We conduct our experiments on a dataset of millions of products for a large e-commerce retailer with the objective of producing multi-horizon forecasts for the next 52 weeks.

Table 1: We show the final WQL for P50 and P90, and the combined WCRPS on the two validation datasets DB and M5. Goldilocks with temperature 2 performed best on M5, the out-of-sample validation dataset. DIS rank second on M5, as expected, given that both have a similar behavior. We put in bold the best two of each column.

	DB			M5		
	P50	P90	WCRPS	P50	P90	WCRPS
Uniform	**0.4586**	**0.7060**	**0.5328**	0.4190	0.5646	0.4626
SIS	0.5216	0.8356	0.6158	0.4485	0.5772	0.4871
DIS	0.7479	0.9985	0.8231	**0.4168**	**0.5529**	**0.4577**
Goldilocks 1.0	**0.5106**	**0.8199**	**0.6034**	0.4512	0.5852	0.4914
Goldilocks 1.5	0.5942	0.9718	0.7075	0.4566	0.6382	0.5111
Goldilocks 2.0	0.5844	0.8812	0.6734	**0.3989**	**0.5119**	**0.4326**

Algorithm 1 Using Equation (1) to update $p(s)$ corresponds to DIS; whereas using Equation (2) corresponds to Goldilocks. Skipping the update $p(s)$ corresponds to SIS, and additionally using a constant in place of the loss corresponds to Uniform. The hyperparameter α controls the moving average decay.

```
Segment products by velocity in equal total demand
d_s ← velocity for each segment s
w_s ← d_s · WCRPS_0                          ▷ scaled loss for zero
while training do
    s ← segment randomly sampled using p(s)
    a ← product uniformly sampled from segment s
    ω'_a ← d_s · WCRPS_a                     ▷ scaled loss after one step
    w_s ← αw_s + (1 − α)ω'_a
    p(s) ← probability updated with w_s
end while
```

The models are trained on four years of demand data from this dataset, and evaluated on one year held out for validation. The features used in each are similar to Eisenach et al. [4], Quenneville-Belair et al. [7], Wolff et al. [11]. They include static (e.g. product catalog fields), historical (e.g. past demand), and future (e.g. promotions). We call the training part of this dataset DT, and the validation and backtesting part, DB.

We also validate on M5 [5, 6], a time series public dataset consisting of sales data covering a time period outside of the training window. It contains about 68 million sales for 3049 items across 10 stores in the US with date ranging from 2011-01-29 and 2016-06-19. For these experiments, we aggregate complete weeks from the week start on Sunday 2011-01-30 to the week starting on Sunday 2016-06-12. We use one year as evaluation, and refer to this out-of-sample validation dataset as M5.

We compare MQ-CNN trained for 20 thousand steps, with Uniform, SIS, DIS, and Goldilocks. Uniform simply takes a segment uniformly at random, and then a product from that segment uniformly at random. For Goldilocks, we experimented with three temperatures: 1.0, 1.5, and 2.0. For each, we use 100 segments for our experiment, that are automatically assigned so as to give about the same total number of units of demand per segment, in the training window. We use $\alpha = 0.1$. We show in Table 1 a comparison of the final validation losses, see Appendix C for the training trajectories. We find that Goldilocks with temperature 2 performed best on M5, the out-of-sample validation dataset. DIS rank second on M5, as expected, given its similar behavior.

4 Conclusion and Future Work

We introduced a new active sampling bandit called Goldilocks that samples across multiple tasks, here corresponding to difference velocity groups of products, based on samples that are not too hard, not too easy, but just right to learn. We saw that the out-of-sample validation error on M5 was lowest for Goldilocks with temperature 2. DIS ranked second on M5, as expected given its similar behavior of dynamically putting more emphasis on segments with larger errors.

For now, we assumed that ergodicity holds – where the time average of one individual (or product) is equivalent to the ensemble average across many individuals (or products). This allowed us to focus on sampling across products with their full history as is done with forking sequences [3, 4, 7, 11], instead of sampling products and time windows. To test the ergodicity assumption, a natural extension of this work is to update the segmentation in a way that orders the products first by their seasonality or price bands, then by their total demand, and then continuing as before.

References

[1] J. Berrisch and F. Ziel. Crps learning. Journal of Econometrics, 237(2):105221, Dec. 2023. ISSN 0304-4076. doi: 10.1016/j.jeconom.2021.11.008. URL http://dx.doi.org/10.1016/j.jeconom.2021.11.008.

[2] J. Berrisch and F. Ziel. Crps learning. Journal of Econometrics, 237(2):105221, Dec. 2023. ISSN 0304-4076. doi: 10.1016/j.jeconom.2021.11.008. URL http://dx.doi.org/10.1016/j.jeconom.2021.11.008.

[3] K. C. Chen, L. Dicker, C. Eisenach, and D. Madeka. Mqtransformer: Multi-horizon forecasts with context dependent attention and optimal bregman volatility. 2022. URL https://www.amazon.science/publications/mqtransformer-multi-horizon-forecasts-with-context-dependent-attention-and-optimal-bregman-volatility.

[4] C. Eisenach, Y. Patel, and D. Madeka. Mqtransformer: Multi-horizon forecasts with context dependent and feedback-aware attention, 2022. URL https://arxiv.org/abs/2009.14799.

[5] A. Howard, inversion, S. Makridakis, and vangelis. M5 forecasting - accuracy. https://kaggle.com/competitions/m5-forecasting-accuracy, 2020. Kaggle.

[6] S. Makridakis, E. Spiliotis, and V. Assimakopoulos. M5 accuracy competition: Results, findings, and conclusions. International Journal of Forecasting, 38(4):1346–1364, 2022. ISSN 0169-2070.

doi: https://doi.org/10.1016/j.ijforecast.2021.11.013. URL https://www.sciencedirect.com/science/article/pii/S0169207021001874. Special Issue: M5 competition.
[7] V. Quenneville-Belair, M. Wolff, B. Willhelme, D. Madeka, and D. Foster. Distribution-free multi-horizon forecasting and vending system. 2023. URL https://www.amazon.science/publications/distribution-free-multi-horizon-forecasting-and-vending-system.
[8] A. Rutherford, M. Beukman, T. Willi, B. Lacerda, N. Hawes, and J. Foerster. No regrets: Investigating and improving regret approximations for curriculum discovery, 2024. URL https://arxiv.org/abs/2408.15099.
[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.
[10] R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka. A multi-horizon quantile recurrent forecaster, 2018.
[11] M. Wolff, K. Olivares, B. Oreshkin, S. Ruan, S. Yang, A. Katoch, S. Ramasubramanian, Y. Zhang, M. Mahoney, D. Efimov, and V. Quenneville-Belair. Spade: Split peak attention decomposition. 2024. URL https://www.amazon.science/publications/spade-split-peak-attention-decomposition.

Appendix A Mean

To recover the mean from an inverse of the cumulative density function (CDF, the plot of Percentiles vs Quantiles), which we denote $F(x)$. As noted in [1], we have the following identity

$$E[x] = \int_0^\infty (1 - F(x))\,dx - \int_{-\infty}^0 F(x)\,dx = \int_0^\infty (1 - F(x))\,dx = \int_0^1 F^{-1}(y)\,dy$$

where we used integration by parts, non-negativity of the distribution, and $F(0) = 0$. Therefore, to recover the expectation of an interpolated distribution, we compute the area under the curve of the inverse CDF (the plot of Quantiles vs Percentiles). To approximate the mean when it exists, we can then proceed with numerical integration.

For example, we could compute the sum of quantiles at equally spaced percentiles from 0 to 0.99, and divide by 100 to get an approximation of the mean. This would thus assign to each quantile a weight of 1/100. Asymptotically, as the number of percentiles increase, the approximation converges to the mean.

The weights mentioned in Section 1 (e.g. 70%, 30%) can be used to sum the quantiles directly and provide an approximate forecast of the mean.

Appendix B Goldilocks Temperature

The maximum of Equation (2) occurs at the midpoint between the roots,

$$\frac{\gamma^{-1} w_{\min} + \gamma w_{\max}}{2}.$$

To find $\gamma \geq 1$ making the maximum occur at $w_{\max}$, we solve

$$w_{\max} = \frac{\gamma^{-1} w_{\min} + \gamma w_{\max}}{2}$$

and get

$$\gamma = 1 + \sqrt{1 - w_{\min}/w_{\max}}.$$

We recall that $0 \leq w_{\min} < w_{\max}$, and, if $w_{\min}$ is much less than $w_{\max}$, then $\gamma \approx 2$.

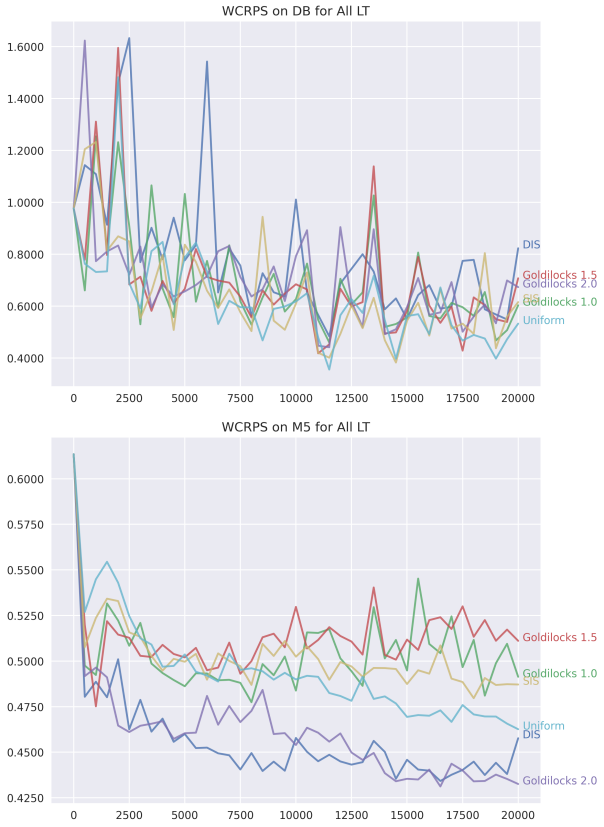

Figure 1: The training trajectories of the WCRPS for the validation datasets, DB and M5, confirm the best two of each dataset from Table 1.

Appendix C Training Trajectories

We show in Figure 1 the training trajectories of the validation WCRPS on both DB and M5 for SIS, DIS, and Goldilocks with temperature 1.0, 1.5, 2.0.