Autoforecaster: A library for automatic time series structure discovery and forecasting

Jonathan P. Chen*†, Neeraj Pradhan†, Zhen Cao†, Siddarth Chandar, Ricky Liang

†Meta

Abstract

We present Autoforecaster*, an open-source PyTorch library for forecasting time series using Gaussian Processes (GPs) with flexible kernel compositions with changepoint detection and kernel search. GPs are a powerful Bayesian approach for modeling time series, providing both point predictions and uncertainty estimates. A key challenge in applying GPs is selecting an appropriate kernel to capture the covariance structure of the data. Our library addresses this by implementing a fixed kernel composition approach, where the kernel is constructed as a sum of base kernels, each capturing different aspects of the data (e.g. trend, periodicity, noise). The library also provides functionality for kernel search, automatically optimizing the hyperparameters of the kernel to maximize the marginal likelihood of the data. By combining these techniques, the library simplifies the process of applying GPs to time series forecasting. We validate Autoforecaster on several Kaggle time series datasets, demonstrating improved accuracy compared to baseline methods. The library is available under an open-source license, providing a valuable tool for researchers and practitioners working with time series data.

Introduction

Time series challenges, seen in fields like finance and global weather predictions, require a solid grasp of structure in data for effective forecasting [12]. Traditionally, understanding these structures involved complex and error-prone bespoke data science, relying on manual data fitting [19]. Responding to the need for simpler and automated approaches, researchers developed methods to streamline this process. In our study, we propose a new method by combining automatic change point detection with a Gaussian Process with a composite kernel from Corani et al. (2020). This approach competes effectively with manual methods and fully automatic forecasting, providing accuracy and potential time savings in specific scenarios [12]. Our research marks progress in making time series analysis more accessible and efficient, offering insights for researchers in this complex field.

Our exploration involves combining Gaussian Processes (GPs) with change point detection methods and assessing their performance against manual forecasting and fixed kernel composition (FKC) baselines. GPs are adept at handling uncertain or noisy data, providing valuable predictions where confidence is crucial [16]. Kernel selection allows for precise fitting, capturing specific trends, and modeling complex relationships. The choice of suitable kernels and determining similarity between data points is crucial for constructing effective GP models [16].

Manual forecasting, relying on human expertise, is a longstanding method for predicting trends but has drawbacks due to its subjective nature and time-intensive process [3]. In contrast, if we can automate the kernel selection process, we can make time series forecasting more powerful and accessible to data scientist and software engineers [19]. By combining a fixed kernel composition with change point detection, our newest model excels at identifying specific points in a time series where significant statistical changes occur [5]. Unlike using only a FKC, modeling change points dynamically recognizes shifts in trends, improving

*correspondence to: jpchen@meta.com

*`http://github.com/jpchen/autoforecaster`

overall model fit [5]. We show on the Kaggle M4 and M5 forecasting competitions that for comparable run times, our automatic model is competitive with manual methods [1].

Gaussian Processes for Time Series

A time series problem involves predicting future values of a variable based on its past observations, where each observation is recorded at discrete time steps.

Time series data can be represented as follows:

$$X = (x_1, ..., x_n)$$

$$Y = (y_1, ..., y_n)$$

where x_i represents the observation at time step i in the input sequence, and y_i represents the corresponding output. The forecasting problem involves predicting future values y_{n+h} which involves computing:

$$p(y_{n+1}, ..., y_{n+h}|y_1, \ldots, y_n)$$

Gaussian Process

A GP is a set of random variables such that the joint distribution of any finite subset of these variables follows a Gaussian distribution. We can think of a GP as a distribution over functions, where our goal is to learn a posterior over all possible functions that can explain the data.[16]. A GP is fully defined by a mean and covariance function (kernel).

The mean function in GPs can take various forms. A commonly used choice is a constant $\mu(x) = \mu$. When fitting a GP model to data, μ is often estimated using the available data. The mean function $\mu(x)$ of a Gaussian process is given by the expected value of the process at a specific point x [6]:

$$\mu(x) = \mathbb{E}[f(x)]$$

We use a constant mean function in our paper, and rely on the kernels below to capture the data patterns.

Covariance Functions

The kernel determines the strength of the correlation between points [6]. Mathematically, the covariance function $k(x, x')$ is defined as the expected value of the product of the differences between the target values and their respective mean values [6]:

$$k(x, x') = \mathbb{E}[(f(x) - \mu(x))(f(x') - \mu(x'))]$$

Kernel selection and discovery is an open area of research. The most common kernel is the RBF kernel, also known as the squared exponential kernel:

$$K_{\text{RBF}}(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2l^2}\right) \tag{1}$$

This kernel is effective for modeling non-linear trends, which computes the distance between pairs of data points in a feature space. The lengthscale parameter l here determines how relevant the feature is for the learned function, which is visually represented by the smoothness of the functions. Longer lengthscale will yield smoother functions, while shorter lengthscale yields rougher functions. It is worth noting that a shorter lengthscale may bring the risk of overfitting, i.e., the model is highly fitted to the existing data and performs poorly on new data.

The Matérn kernel, which is interestingly the generalization of the RBF kernel although it appears less frequently, is:

$$K_{\text{Matern}}(x, x') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}\|x - x'\|}{l} \right)^{\nu} K_{\nu} \left(\frac{\sqrt{2\nu}\|x - x'\|}{l} \right) \tag{2}$$

This kernel has an additional parameter ν, which controls the smoothness of the function. The smaller ν is, the less smooth the function is. As $\nu \to \infty$, the kernel becomes equivalent to the RBF kernel.

Linear kernel, a simpler kernel that captures linear trends, is:

$$K_{\text{Linear}}(x, x') = x^T x' \tag{3}$$

It is best used to capture linear relationships between data points but is ineffective when dealing with non-linear ones. It does not have a lengthscale parameter.

Another common kernel is the Periodic (PER kernel):

$$K_{\text{Periodic}}(x, x') = \exp\left(-\frac{2\sin^2(\pi\|x - x'\|/p)}{l^2} \right) \tag{4}$$

Most time series often consist of periodic patterns, whether daily, monthly, or annually, and a periodic kernel can capture the repeated trends within the data. The lengthscale parameter l here operates similarly to the lengthscale parameter in the RBF kernel, where a longer length scale decreases the local variation within repetition and captures less detail.

The Rational Quadratic (RQ) kernel is defined as:

$$K_{\text{RQ}}(x, x') = \left(1 + \frac{\|x - x'\|^2}{2\alpha l^2} \right)^{-\alpha} \tag{5}$$

This kernel can be interpreted as an infinite number of different RBF kernels with different lengthscales added together. This makes it suitable for dealing with complex data sets that don't have a simple pattern. The lengthscale parameter here controls the spread of the covariance. It has a positive correlation, while the scale-mixture manipulates the number of local variations, increasing the scale-mixture and reducing local variations.

The Spectral Mixture (SM) kernel is:

$$K_{\text{SM}}(x, x') = \sum_{k=1}^{K} w_k \cos(2\pi f_k \|x - x'\| + \phi_k) \exp\left(-2\pi^2 f_k^2 l^2 \|x - x'\|^2 \right) \tag{6}$$

This kernel is powerful because it learns all the frequencies in the data by learning the spectral density. The spectral density indicates the probability density of the corresponding kernel or how likely each frequency is in the data. Hence, the SM kernel can capture extremely complex patterns.

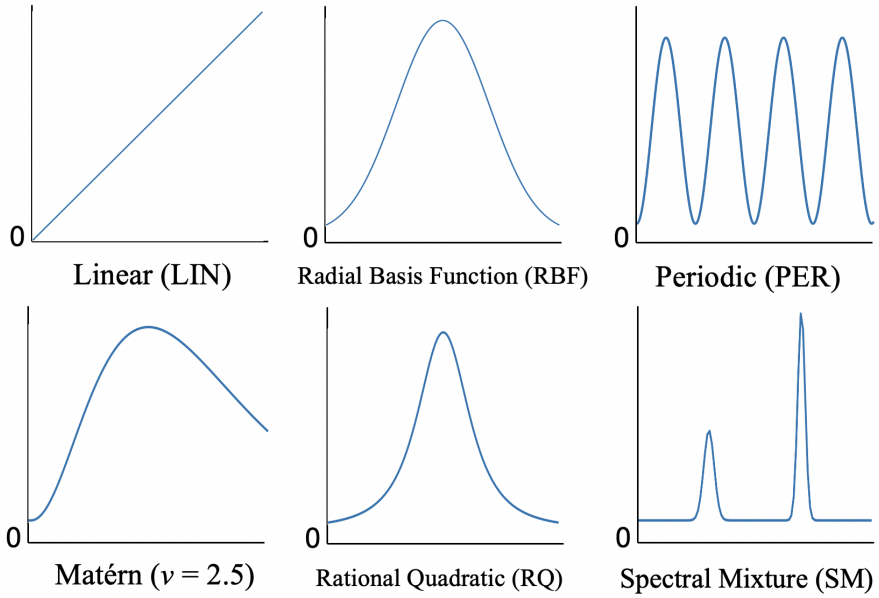

Figure 1: Functions that GP kernels model.

DataTensor

We introduce the `DataTensor`, a PyTorch tensor with Pandas semantics that stores additional metadata to facilitate time series modeling. `DataTensor` wraps `torch.Tensor` object and associates header information specifying column names for the innermost dimension of the tensor. This allows maintaining the semantic meaning of the data and enables intuitive column-based indexing and operations. The `DataTensor` class provides normalization functionality, allowing scaling of specific columns to zero mean and unit variance. Columns to normalize can be specified either as a list of column names or a dictionary mapping column names to corresponding `torch.distributions.Transform` instances. To ensure seamless integration with PyTorch, the `DataTensor` class overrides various PyTorch functions and operators. It properly handles tensor operations, indexing, and slicing, preserving the associated metadata and ensuring consistency across operations. The class provides utility functions for converting pandas DataFrames to `DataTensor` instances, facilitating integration with existing data processing pipelines.

Models

Manual Forecasting

We refer to "manual forecasting" as the trial and error process of fitting a model (in our case a GP kernel) to data. In this approach, researchers carefully analyze the data to discern trends and try suitable kernels. The process of kernel composition then takes place, involving the addition or multiplication of kernels to construct a model capable of capturing a diverse range of patterns. This is particularly advantageous when dealing with data exhibiting complex functions.

Similar to architecture search in deep learning, the resulting fit is a local optima conditioned on the quality of the kernels the modeler selects. It is still the predominant method of fitting short term time series since researchers can make minute adjustments at every step of the modeling process. Unlike automated methods, researchers may possess information not present in the data, enabling them to make more comprehensive predictions. Though time consuming, their unique understanding of the data may yield unexpected outcomes that automated predictions cannot replicate.

Fixed Kernel Composition

Fixed kernel composition (FKC) is using a single fixed kernel for all problems of a certain characteristic, eg time series data. Instead of manually selecting and adjusting kernels to construct forecasting models, we use a single fixed kernel for all the data sets as in [8]. The biggest change is removing the PER kernel where no seasonal pattern was observed in the M4 data.

$$\mathrm{K_{M4}} = \mathrm{RBF} + \mathrm{SM1} + \mathrm{SM2}$$
$$\mathrm{K_{M5}} = \mathrm{RBF} + \mathrm{SM1} + \mathrm{SM2} + \mathrm{PER}$$

Corani et al. also adopted a hierarchical Bayesian perspective to FKC, treating the hyperparameters of each kernel as random variables and learning a distribution for each one. This allowed them to quantify the uncertainty associated with the hyperparameter and potentially improve the model's robustness to overfitting. Instead, we employ maximum likelihood estimation (MLE) to learn a single point estimate of the hyperparameter that best explains the observed data. While this method doesn't explicitly account for uncertainty, it is computationally efficient and performed sufficiently well, particularly with the larger datasets we used. While automatic, fixed kernel composition may lack the depth of human insights and domain-specific knowledge that manual forecasting offers, potentially limiting accuracy. Finally, relying solely on statistical measures in automation also raises the risk of overfitting, hindering adaptability to new data.

Change point Detection

Change point detection is a method used to identify points in a time series where abrupt and significant variations have occurred [2]. We use binary segmentation as described below to for change points, and then fit RBF kernels on the each window of data bifurcated by the changepoints.

$$K_{\text{input\_change}}(x_1, x_2; \{\lambda, \sigma_1, \sigma_2\}) = \lambda^2 \exp\left(-\frac{1}{2\sigma_2^2}|x_1 - x_2|\right), \quad \text{for } x_1 \geq \text{change point} \tag{7}$$

Binary Segmentation

We use a greedy binary segmentation method to identify points in a sequence of data where a significant change or "break" occurs, which is the change point. It first detects a change point within the entire time series, then the time series is split into two around this change point, and the detection process will be repeated on the two resulting segmented time series. This process is repeated until no further change point is detected [18].

The initial change point, denoted as $\hat{t}^{(1)}$, is given by:

$$\hat{t}^{(1)} := \arg\min_{1 \leq t < T-1} [c(y_{0..t}) + c(y_{t..T})]$$

where $c(\cdot)$ is the cost function. This operation is "greedy," searching for the change point that minimizes the sum of costs the most. The signal is then split into two at the position of $\hat{t}^{(1)}$. This process is repeated on the resulting sub-signals until a stopping criterion is met. The algorithm's complexity is of the order $O(T \log T)$, making it efficient but approximate.

Since the change points separate time series into segments with similar characteristics, it can more accurately and precisely capture the trend and use it to forecast future events. This becomes especially practical in real-world applications where data are constantly affected by external factors that lead to abrupt changes within the dataset. Change points also segment data into different periods by characteristics, giving researchers deeper insights into understanding the properties and behaviors of each segment, potentially allowing businesses to make better decisions.

Figure 2: BinSeg Algorithm

Require: Signal $\{y_t\}$, T, cost function $c(\cdot)$, stopping criterion.
Initialize $L \leftarrow \{\}$. ▷ Estimated breakpoints.
repeat
  $k \leftarrow |L|$. ▷ Number of breakpoints
  $t_0 \leftarrow 0$ and $t_{k+1} \leftarrow T$. ▷ Dummy variables.
  if $k > 0$ **then**
    Denote by t_i (for $i = 1, \ldots, k$) the elements in ascending order of L, i.e., $L = \{t_1, \ldots, t_k\}$.
  end if
  Initialize G as a $(k+1)$-long array. ▷ List of gains
  for $i = 0, \ldots, k$ **do**
    $G[i] \leftarrow c(y_{t_i}..t_{i+1}) - \min_{t_i < t < t_{i+1}}[c(y_{t_i}..t) + c(y_t..t_{i+1})]$.
  end for
  $b_i \leftarrow \arg\max_i G[i]$.
  $\hat{t} \leftarrow \arg\min_{t_{b_i} < t < t_{b_i+1}}[c(y_{t_{b_i}}..t) + c(y_t..t_{b_i+1})]$.
  $L \leftarrow L \cup \{\hat{t}\}$.
until stopping criterion is met.
Ensure: Set L of estimated breakpoint indexes.

Experiments

The datasets we have chosen for this experiment are snippets of data from the M4 [13] and M5 [14] datasets, which are datasets used in the series of the Makridakis competition that evaluate and compare different forecasting methods. We select 1871 time series in the M4 dataset with 1853 data points for training and 18 data points for testing, and 1000 time series in the M5 dataset with the first 900 data points for training and the last 100 data points for testing. We applied 3 different forecasting approaches to both datasets: manual kernel construction, fixed kernel composition, and change point detection. The first two approaches are the baselines, while we are particularly interested in the results of the change point detection approach. We used GPytorch [11] and the autoforecasting library [7] with the following manual kernels to implement and run the experiments.

$$\mathrm{K_{M4}} = \mathrm{RBF} \times \mathrm{PER} + \mathrm{RQ}$$
$$\mathrm{K_{M5}} = \mathrm{RBF} \times \mathrm{PER} + \mathrm{MAT}$$

Multiplying the RBF and Periodic kernels and adding the result to the RQ kernel creates a composite kernel structure. This combination allows the model to capture long-term and periodic patterns in the data through the RBF-Periodic component while accounting for more complex and irregular relationships with the RQ component. This combination is advantageous because it provides a flexible way to model data that exhibits both smooth long-term trends and periodic behavior and enhances the model's robustness to noisy data and irregularities in the data, using the detailed set of kernels in the specific combination, the GP model was trained with about 98-99 percent of the selected M4 row, and the rest of the data was used for testing, with 1000 iterations of training being used for the model.

Metrics

Mean Absolute Error (MAE) is a metric commonly used to evaluate the accuracy of predictive models, particularly in regression problems. It measures the average absolute difference between the predicted values ($\hat{y}_t$) and the true values (y_t) as follows:

$$MAE = \frac{1}{T}\sum_{t=1}^{T}|y_t - \hat{y}_t| \tag{8}$$

Here, T represents the number of data points or time instances in the test set. A lower MAE indicates a

better fit of the model of the data, as it quantifies the average magnitude of errors between predictions and actual values.

Continuous-Ranked Probability Score (CRPS) is a metric used to assess the accuracy of probabilistic forecasts, such as predictive distribution functions. It compares these probabilistic forecasts with the observed data. CRPS is defined as follows:

$$CRPS(F_t, y_t) = -\int_{-\infty}^{\infty} (F_t(z) - \mathbf{1}_{z \geq y_t})^2 \, dz \tag{9}$$

Here, $F_t(z)$ represents the cumulative predictive distribution function at time t, and y_t is the observed target value at time t. The indicator function $\mathbf{1}_{z \geq y_t}$ is 1 for $z \geq y_t$ and 0 otherwise. A lower CRPS value indicates a better fit of the probabilistic forecast to the actual observations.

Log-likelihood (LL) is a measure used to evaluate how well a statistical model describes the observed data, particularly in the context of models providing probabilistic predictions. The LL for a test set is defined as follows:

$$LL = \frac{1}{T}\left(-\frac{1}{2}\sum_{t=1}^{T}\log(2\pi\sigma_t^2) - \frac{1}{2\sigma_t^2}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2\right) \tag{10}$$

Here, T is the number of data points in the test set, σ_t^2 represents the variance of the predictive distribution at time t, y_t is the observed value, and $\hat{y}_t$ is the predicted value. A higher LL indicates a better match between the model's predictions and the observed data, as it represents the likelihood.

Our study employed three distinct forecasting methods—manual forecasting, FKC, and change point—on randomly selected time series from the M4 and M5 datasets. Each method was for 1,000 iterations, utilizing 90% of the data, and 10% withheld for test. We measured MAE, CRPS, LL, and CPU run times averaged over five runs.

Results

Analyzing the change point model's performance on the M4 dataset reveals the model's observations. The model's confidence bounds are larger than the manual model's and FKC's. Yet, they encapsulate the dataset's dynamics without succumbing to the overfitting of the dataset seen with the FKC model. This suggests flexibility in the model's representation of general trends and avoids an overly close adherence to noise or outliers.

Examining the shape of the confidence bounds and their alignment with the test mean (4) indicates that the model adeptly reflects the underlying structure of the M4 data. This ability to capture data dynamics without overfitting distinguishes the model from FKC counterparts, which might tend to closely follow training data at the risk of losing generalization. Regarding test data predictions, the model follows the overall trend of the M4 data but falls short in capturing a final small upward trend present in the actual training data. This discrepancy is reflected in an increasing variance between the predicted test mean and the real test data, highlighting a limitation in the model's ability to forecast subtle shifts in data and make better general trend predictions.

	Manual	**FKC**	**FKC + CP**
MAE	$\mathbf{0.0852 \pm 0.0168}$	0.412 ± 0.0115	0.1563 ± 0.0295
CRPS	0.084 ± 0.0051	0.3124 ± 0.0105	$\mathbf{0.0803 \pm 0.0087}$
NLL	$\mathbf{0.2916 \pm 0.0077}$	0.7544 ± 0.0090	0.3142 ± 0.0221
Average CPU Time Per Iteration	$\mathbf{7.32 \pm 0.31}$	25.65 ± 0.94	8.71 ± 0.84

Table 1: M4-Related Numerical Results: Averaged Over Five Runs

A noteworthy aspect of the change point model's behavior is its handling of outliers. Instead of becoming confused or overcompensating for these anomalies, the model strategically uses them. Outliers are incorporated to subtly adjust the training mean, guiding it towards a more comprehensive fit that considers the

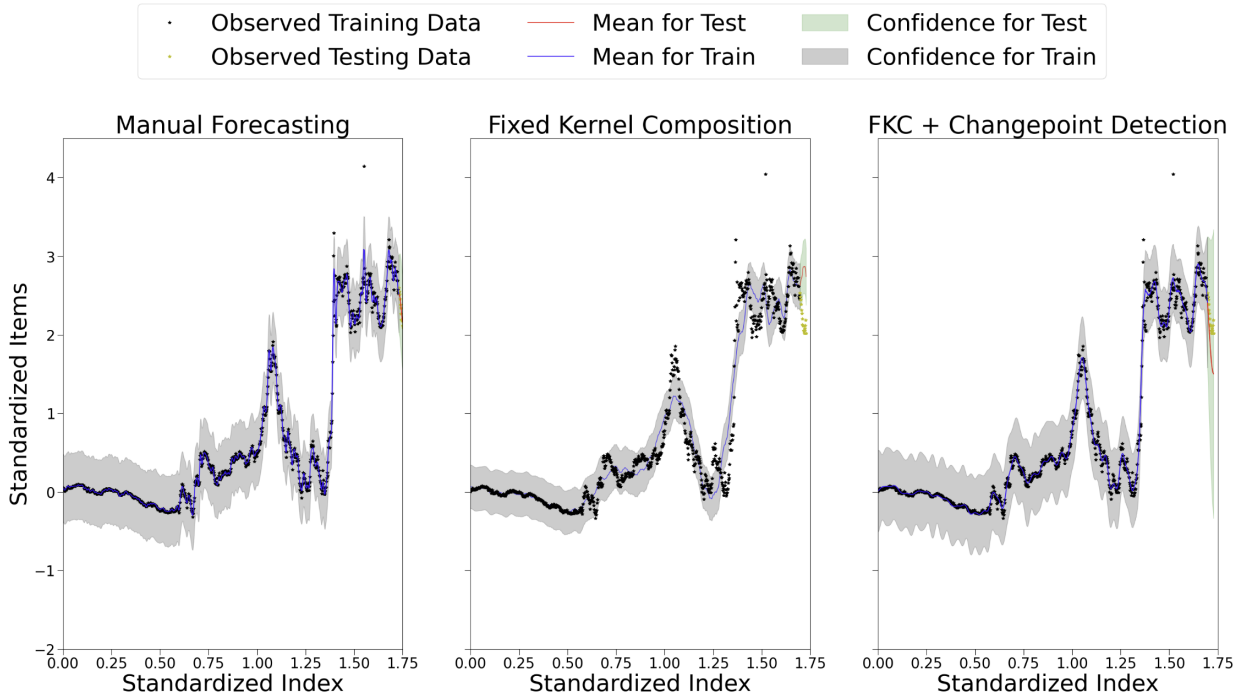

Figure 3: This is the overview of the M4 dataset that analyzes anonymous monthly sales figures, showing training and testing means calculated by all three models, along with the data used for training and testing, and the confidence bounds of each model.

entire dataset, including outliers. This approach indicates adaptability without succumbing to the influence of outliers that might lead to overfitting.

In contrast to the M4 dataset, the Change-Point detection model encountered challenges when applied to the more periodic M5 dataset. The periodic patterns inherent in M5 seemed to elude the model, highlighting its inability to discern and adapt to the intricacies of such cyclical variations (5). This contrast is noteworthy as the less periodic M4 dataset showcased a more favorable performance in analyzing specific trends of at least parts of the data.

One discernible observation was the model's efficacy in capturing the general trend in transitioning from the training to the test phase in the M5 dataset (6). This ability to grasp overarching trends indicates a certain level of adaptability in understanding the broader patterns within the data. However, this proficiency came at the cost of neglecting the finer periodic trends that characterize the M5 dataset.

A difference emerged when comparing the confidence bounds between the M4 and M5 results. The M5 dataset exhibited smaller confidence bounds, indicative of a more conservative approach in the model's predictions or better confidence in its model's training and testing means. This conservatism could be attributed to the model's inclination towards generalizing the data patterns at the expense of accommodating the pronounced periodicity in the M5 dataset.

Further scrutiny of the M5 data revealed variations aligning with relative outliers, particularly at the maxima or minima of the apparent "periods" in the dataset. These variations might correspond to critical points in the periodic cycles. The Change-Point detection model, however, seemed to overlook periodic nuances, emphasizing a tendency toward a more generalized representation of the data, both for the M4 and M5 dataset results.

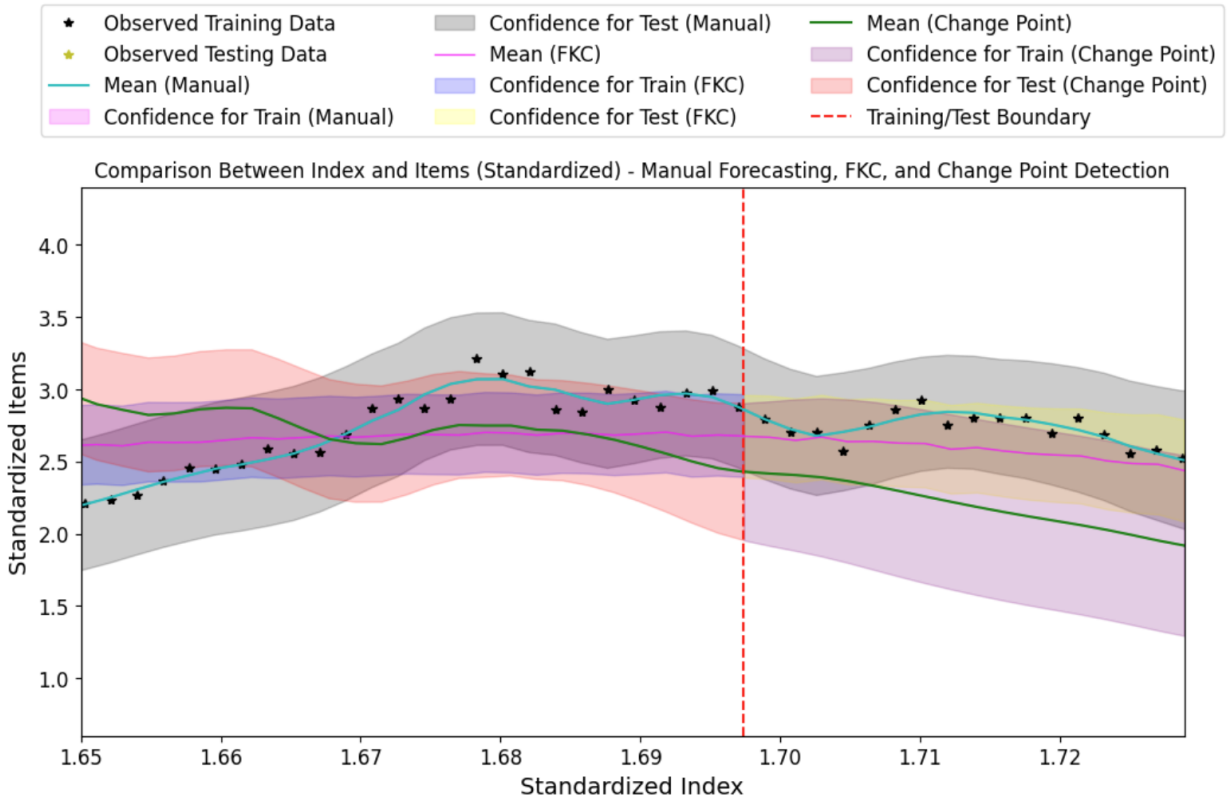

Figure 4: This is the testing portion of the M4 dataset, showing the testing means calculated for each model, along with the test data, and the confidence bounds for each model.

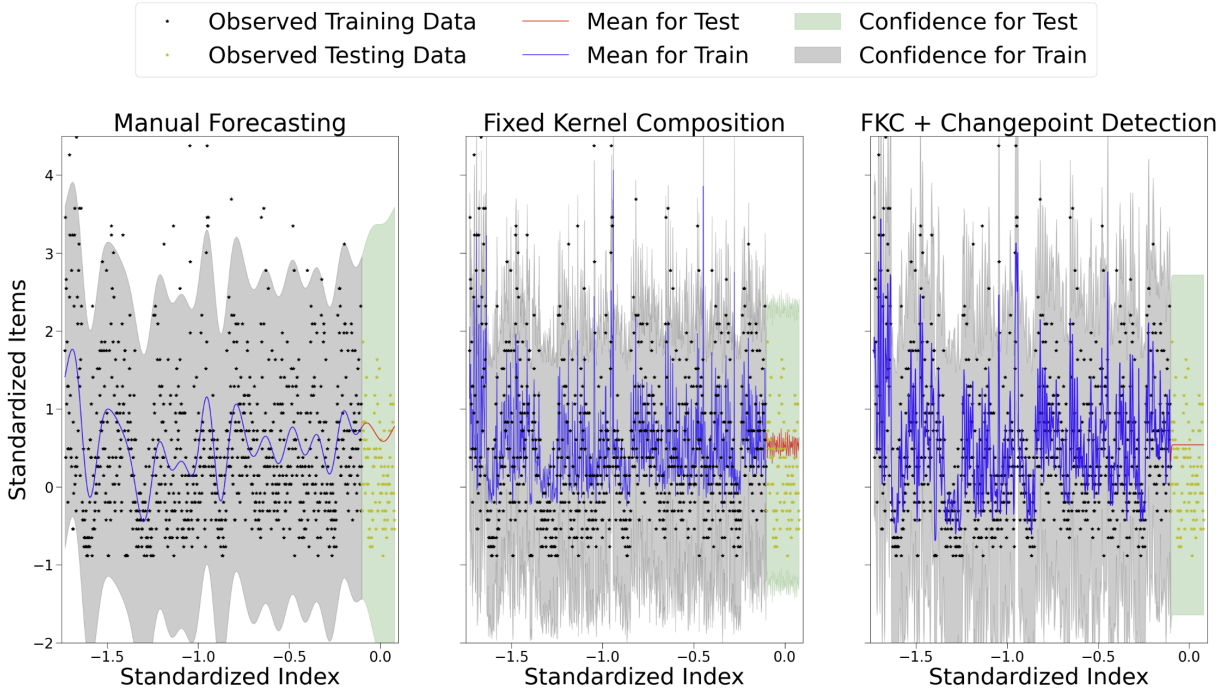

Figure 5: This is the overview of one of the rows of the M5 dataset of sales data from a store in California. The figures shows the training (blue) and testing (red) means along with the confidence bounds of the models.

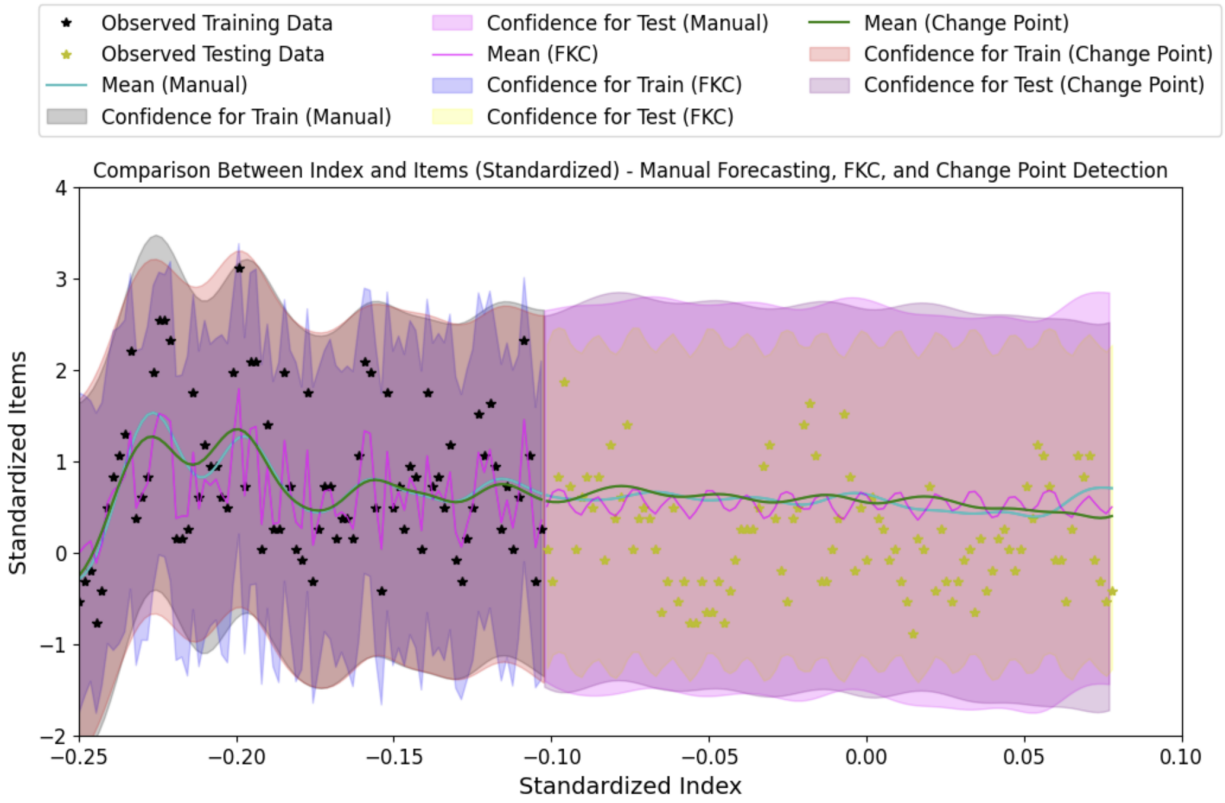

Figure 6: This is the testing portion of the M5 dataset, showing the testing means calculated by each model, along with the test data and the confidence bounds of each model

In our comprehensive analysis of three distinct models applied to both the M4 and M5 datasets, we reviewed their numerical results across four crucial measurement factors of MAE, CRPS, LL (or negative-LL), and CPU time. The models in focus were the manual-fitting mode, the FKC model, and the change point model.

Beginning with the M4 dataset, the manual-fitting model demonstrated superior performance by achieving the best mean and standard deviation combination across the MAE, LL, and CPU run time metrics. However, regarding the CRPS result, the change point model outperformed the others. Notably, change point consistently secured the second position across all other c, approaching the performance of the best model in each metric.

These findings suggest that the manual forecasting model, particularly the variant selected through a meticulous process involving specific kernels, was well optimized for the M4 dataset. Simultaneously, the consistently strong performance of change point across all metrics implies that, for datasets characterized by fewer periodic trends and more extended-scale patterns, change point emerges as an optimal model. This is especially true when considering the time investment required for designing and implementing a custom-made manual-fitting model, which demands substantial development efforts.

Turning our attention to the M5 dataset, a different pattern emerges. FKC takes the lead in terms of MAE and CRPS (comparing mean and standard deviations once again), showcasing its effectiveness in capturing the complexities of the dataset. Manual forecasting excels for the negative Log Likelihood (used in this case instead of LL for the M4 results). Surprisingly, change point outshines the others regarding CPU time, considering it was in the middle for the relatively less complicated M4 dataset. In this dataset, FKC is the most successful, closely followed by change point, while manual forecasting lags noticeably behind.

These results highlight the strength of FKC, despite its relatively slower run time due to its fully automated modeling system. Its ability to yield the best results in the periodic M5 dataset, characterized by

	Manual	**FKC**	**FKC + CP**
MAE	0.6186 ± 0.0256	$\mathbf{0.5854 \pm 0.0376}$	0.5976 ± 0.0393
CRPS	0.419 ± 0.00587	$\mathbf{0.4184 \pm 0.0172}$	0.4232 ± 0.0165
NLL	1.2588 ± 0.00594	0.9872 ± 0.04	$\mathbf{0.9868 \pm 0.0364}$
Average CPU Time Per Iteration	$\mathbf{2.06 \pm 1.06}$	3.36 ± 1.20	2.45 ± 0.07

Table 2: M5-Related Numerical Results: Averaged Over Five Runs

numerous localized and general pattern shifts, underscores its effectiveness. change point, with its adept performance, secures the second position, particularly given the dataset's periodicity and evident general patterns in the data. Finally, despite exhaustive efforts and kernel combinations, manual forecasting falls short of replicating the high-performing results achieved in the M4 dataset when applied to the M5 dataset.

Ultimately, change point emerges as a robust performer when confronted with broader data parameters and certain periodic trends. On the other hand, FKC maintains consistent good performance across diverse data types especially for more periodic data, albeit at the cost of extended run times for FKC models. The manual forecasting approach introduces an element of user-dependent variability. However, in our specific case, we successfully created manual forecasting models that, at the very least, matched the performance levels exhibited by both the FKC and change point models.

Our experiments generally ran successfully in the M4 dataset, while exhibiting errors and flaws in the M5 dataset. For example, the confidence bounds demonstrated the same periodic trend in fixed kernel composition, hence producing a questionable forecast. This may be caused by noisy data in conjunction with a suboptimal kernel composition and inadequate training of hyperparameters, which leads to an inaccurate kernel composition that cannot precisely capture the trend and result in erroneous confidence bounds. The time series selected here are visibly noisy and appear abundant amount of outliers, which may disrupt the data's preexisting pattern, hindering the model's performance to capture the underlying trend. This may indicate insufficient and ineffective data preprocessing before we use them for training, especially in removing outliers and adjusting noise levels.

Related Work

Attempts at creating kernel compositions for GPs is not new. For time series problems, Corani et al. used the fixed kernel composition we compare with but emphasizing the use of hierarchical priors for the hyperparameters. For the class of problems they benchmark against, they show that performance improves with this setup.

Structure discovery with GPs has been an active research area over the last decade[10], [15]. [10] uses a greedy search algorithm while [15] introduced a new structure learning algorithm based on sequential Monte Carlo sampling.

Change point detection for time series is also an active area of research. [4] have already implemented change point detection for the automatic analysis of subtle changes within MRI scans, and it turns out to be much less error-prone than a manual approach by experts. [9] used change point detection to infer the effectiveness of government interventions in COVID-19 by evaluating the forecasted value with the actual reported data. More recently, [17] adopted change point detection to analyze many short time series in waste management.

Conclusion

We compared change point detection and composite kernels with manually selected kernels and showed competitive results in terms of accuracy and runtime on 2 open-source time series data. As a relatively new model, change point detection already demonstrates its practical use in automatic time series forecasting and displays potential for future development. Future directions could include more robust detection of longer periods of change windows and developing benchmark datasets for fairer and more objective experiments with change point detection in forecasting.

References

[1] Kaggle competitions. `https://www.kaggle.com/competitions`.

[2] Samaneh Aminikhanghahi and Diane J. Cook. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51(2):339–367, Sep 2016.

[3] J. Scott Armstrong. Principles of forecasting: A handbook for researchers and practitioners,. 2001.

[4] Marcel Bosc, Fabrice Heitz, Jean-Paul Armspach, Izzie Namer, Daniel Gounot, and Lucien Rumbach. Automatic change detection in multimodal serial mri: application to multiple sclerosis lesion evolution. *NeuroImage*, 20(2):643–656, 2003.

[5] Gerrit J. J. van den Burg and Christopher K. I. Williams. An evaluation of change point detection algorithms, Mar 2020.

[6] Christopher K. I. Williams Carl Edward Rasmussen. *Introduction*. The MIT Press, 2005.

[7] Jonathan P. Chen, Neeraj Pradhan, and Zhen Cao. Autoforecasting library. https://github.com/jpchen/autoforecasting.

[8] Giorgio Corani, Alessio Benavoli, and Marco Zaffalon. *Time Series Forecasting with Gaussian Processes Needs Priors*, page 103–117. Springer International Publishing, 2021.

[9] Jonas Dehning, Johannes Zierenberg, F. Paul Spitzner, Michael Wibral, Joao Pinheiro Neto, Michael Wilczek, and Viola Priesemann. Inferring change points in the spread of covid-19 reveals the effectiveness of interventions. *Science*, 369(6500):eabb9789, 2020.

[10] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search, 2013.

[11] Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[12] Rob J. Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. Otexts, Oct 2013.

[13] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020. M4 Competition.

[14] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m5 competition: Background, organization, and implementation. *International Journal of Forecasting*, 38(4):1325–1336, 2022. Special Issue: M5 competition.

[15] Brian J. Patton Mansinghka, Feras A. Saad. Sequential monte carlo learning for time series structure discovery. Jul 2023.

[16] Carl Edward. Rasmussen. Gaussian processes for machine learning. 2006.

[17] Veronika Smejkalová, Radovan Šomplák, Martin Rosecký, and Kristína Šramková. Machine learning method for changepoint detection in short time series data. *Machine Learning and Knowledge Extraction*, 5(4):1407–1432, 2023.

[18] Charles Truong; Laurent Oudre; Nicolas Vayatis;. Selective review of offline change point detection methods. Mar 2020.

[19] Can Wang, Mitra Baratchi, Thomas Bäck, Holger H. Hoos, Steffen Limmer, and Markus Olhofer. Towards time-series feature engineering in automated machine learning for multi-step-ahead forecasting. In *ITISE 2022*, Basel Switzerland, Jun 2022. MDPI.