

# Understanding Model Drift in a Large Cellular Network

Shinan Liu, Francesco Bronzino<sup>†</sup>, Paul Schmitt<sup>‡</sup>, Nick Feamster,  
 Ricardo Borges<sup>§</sup>, Hector Garcia Crespo<sup>§</sup>, Brian Ward<sup>§</sup>  
*University of Chicago, <sup>†</sup>Université Savoie Mont Blanc,*  
*<sup>‡</sup>Information Sciences Institute, <sup>§</sup>Verizon*

{shinanliu, feamster}@uchicago.edu, francesco.bronzino@univ-smb.fr,  
 pschmitt@isi.edu, {ricardo.borges, brian.ward2}@verizonwireless.com,  
 hector.garcia@verizon.com

## ABSTRACT

Operational networks are increasingly using machine learning models for a variety of tasks, including detecting anomalies, inferring application performance, and forecasting demand. Accurate models are important, yet accuracy can degrade over time due to concept drift, whereby either the characteristics of the data change over time (data drift) or the relationship between the features and the target predictor change over time (model drift). Drift is important to detect because changes in properties of the underlying data or relationships to the target prediction can require model retraining, which can be time-consuming and expensive. Concept drift occurs in operational networks for a variety of reasons, ranging from software upgrades to seasonality to changes in user behavior. Yet, despite the prevalence of drift in networks, its extent and effects on prediction accuracy have not been extensively studied. This paper presents an initial exploration into concept drift in a large cellular network in the United States for a major metropolitan area in the context of demand forecasting. We find that concept drift arises largely due to data drift, and it appears across different key performance indicators (KPIs), models, training set sizes, and time intervals. We identify the sources of concept drift for the particular problem of forecasting downlink volume. Weekly and seasonal patterns introduce both high and low-frequency model drift, while disasters and upgrades result in sudden drift due to exogenous shocks. Regions with high population density, lower traffic volumes, and higher speeds also tend to correlate with more concept drift. The features that contribute most significantly to concept drift are User Equipment (UE) downlink packets, UE uplink packets, and Real-time Transport Protocol (RTP) total received packets.

## 1 INTRODUCTION

Network operators are increasingly relying on machine learning models to perform a variety of network operations tasks,

including anomaly detection [32, 33], performance inference [14] and diagnosis, and forecasting [7, 25]. Yet, even if machine learning models are accurate for specific network management tasks on a particular set of test data, deploying and maintaining the models can prove challenging in practice [35]. A significant operational challenge is *model drift*, whereby a model that is initially accurate at a particular point in time becomes less accurate over time—either due to a sudden change, periodic shifts, or gradual drift over time. Previous work in applying machine learning models to network management tasks has generally trained models on fixed datasets [5, 7, 10, 25, 32–34], demonstrating the ability to predict various network properties or instances at fixed points in time. Yet, a model that performs well at a particular point in time does not necessarily perform well on future data.

Models can become less accurate at predicting target variables for a variety of reasons. One cause is a sudden, drastic change to the environment. For example, the installation of new equipment, a software upgrade, or a sudden change in traffic patterns or demands can cause models to suddenly become inaccurate. One notable instance that exhibited such a sudden shift was the COVID-19 pandemic, an exogenous shock that resulted in significant changes to behavior patterns [23] (e.g., reduced mobility) and traffic demands [20] (e.g., as people suddenly began using home WiFi connections more for specific applications and relying less on mobile data). Another characteristic is periodic changes, whereby a model that is accurate on a particular time window may immediately become less accurate but will exhibit higher accuracy at fixed, periodic intervals in the future. For example, one clear phenomena that we observe is a drift in model accuracy with a seven-day period; diurnal and periodic patterns in network traffic are, of course, both well-documented and relatively well-understood. A new contribution in this paper, however, is to demonstrate that machine learning models also exhibit similar patterns with respect to accuracy.

Parameter	Main Finding
KPI (target)	Drift behaviors are different for different KPIs. (Fig. 2, § 4.2)
Model	Individual KPI predictions drift consistently across different models. (Fig. 2a, § 4.3)
Training set size&period	Consistent drift appears when trained using different training set sizes and periods. (Fig. 3a & 3b, § 4.3)
Morphology	Higher population density leads to stronger drift. (Fig. 4, § 5.1)
Traffic & Performance	The lowest volume tier and the highest throughput tier exhibit the highest drift. (Fig. 5a & 5b, § 5.1)
Retrain frequency	Increasing retrain frequency does not help mitigating the drift. (Fig. 8, § 6)
Prediction distance	Lower prediction distance leads to less drift. (Fig. 9, § 6)

**Table 1:** Summary of experiments that have parameters changed and their findings.

Model drift is also a relatively well-understood phenomenon in machine learning and has been studied in the context of a variety of other prediction problems [15, 21, 31]. To our knowledge, this paper is the first to study the phenomenon in the context of cellular networks. Specifically, we study and characterize model drift in the context of a large cellular network, exploring drift for a variety of cellular key performance indicators (KPIs) using nearly three and a half years of data from a major metropolitan area in the United States. *Data drift* refers to changes in the distribution of features that are input to the model. *Concept drift* refers to changes of relationships between model inputs (i.e., features) and the output (i.e., target prediction). In our context, concept drift also entails unique challenges: as opposed to tasks such as image or text classification, where the semantics of prediction occurs on a fixed object and characteristics of the features change relatively slowly over time [12, 15, 41], predictions of network characteristics occur continuously over time, and predictions are occurring within the context of a system that changes over time due to both gradual evolution and exogenous shocks. Moreover, dynamic signal interference due to environment changes often occur in wireless networks [40], a phenomenon that adds another layer of complexity and novelty with respect to previous studies.

Data drift and concept drift are related, but distinct concepts. For example, a feature or a group of features could experience drift over time without affecting model accuracy: that is, the model could continue to accurately predict the target even if characteristics of the input features are non-stationary. Concept drift can be more serious because it implies that a previously trained model may not accurately

<b>Collection period</b>	Jan. 1st 2018 – May 3rd 2021
<b>Number of KPIs</b>	224
<b>KPIs of interests</b>	Downlink volume Throughput Peak active users RRC est. success S1-U call drop rate
<b>Number of eNBs</b>	898
<b>eNBs by Morph.</b>	(Dense) Urban: 98 Suburban: 496 Rural: 184
<b>Number of logs</b>	737,577

**Table 2:** Summary of dataset.

predict target values because the relationships between the features and the target have changed over time. For example, an increase in traffic volume (a feature) could exhibit drift over time, but it might still accurately affect a target like call drop rate, even as volumes change over time. On the other hand, concept drift could cause the *relationship* between the volume and the call drop rate to change over time; for example, a software upgrade might cause call drop rates to increase or decrease, given traffic volume. Both data drift and concept drift are important to characterize. While understanding concept drift has obvious importance since it relates to model accuracy, understanding data drift is important because it helps explain concept drift.

This paper both introduces new findings concerning the best ways to adapt to model drift and provides general methods and approaches to help operators recognize and mitigate model drift in their networks. For example, a common practice to combat drift in large networks is to regularly retrain models [19, 37]. However, in many cases, this common practice turns out to be the wrong approach: due to the periodic nature of certain components of drift, continually retraining using a fixed-size window of only recent observations can actually degrade model performance.

We make contributions in three areas; Table 1 summarizes the major results and where to find them in the paper. First, we explore and characterize *data drift* from the input KPIs in a large cellular network. We first characterize different types of data drift, highlighting four main types of drift: (1) gradual; (2) incremental; (3) sudden; (4) recurring context. Gradual and incremental drift occur as a result of changes in distributions of variables over time. Incremental drift is similar to

gradual drift but occurs at more discrete intervals, as opposed to continuously. Sudden drift can occur due to exogenous shocks that could result from either changes to deployed infrastructure, changes to traffic patterns, or a combination of factors. Drift resulting from recurring context can occur for a variety of reasons; one reason is seasonality, as weather, foliage and other factors affect radio propagation with regularity. Our study highlights drift due to recurring context at various periodic intervals; weekly recurring contexts are particularly prevalent.

Second, we demonstrate *concept drift* in a large cellular network comparing different KPIs, model families, training set sizes, and periods across a wide range of prediction models and prediction tasks. We find that concept drift occurs across a wide range of KPIs (i.e., volume, throughput, call drop rate, peak user), and that drift occurs across a wide variety of machine learning models, and families of models. In other words, drift occurs regardless of the choice of model. We also find that model drift occurs consistently and independently of both the size and period of the training set.

Third, we apply signal processing techniques including frequency decomposition, and machine learning-based analysis including feature importance movements and contributing factor explanations to illuminate the causes of model drift. Although this paper does not go so far as to build a mechanisms to detect or mitigate model drift, our analysis lays the groundwork for techniques that could help operators better mitigate drift in large operational networks. We examine the effects from various contributing factors to drift and apply permutation-based feature importance to see which feature set contributes most to concept drift.

## 2 RELATED WORK

Model drift is a pervasive phenomenon in machine learning that has been well-studied in various contexts and scenarios. In this section, we first summarize the efforts of previous work on model drift. We then present related studies in the particular context of networking.

Concept drift has attracted considerable attention across many real-world contexts. Changes in data yield the phenomenon of concept drift, due to dynamic and non-stationary environments [21, 31]. For example, spam detection faces challenges concerning drift, as spammers are actively changing the underlying distribution of messages [12]. More generally, machine learning in security contexts face problems dealing with evasion. Recommender systems introduce another context where drift occurs: user-side effects such as tastes and preferences and item-side effects such as popularity both change over time [15, 22]. Similar situations occur in monitoring systems [29], fraud detection [41], and malware detection [6, 18, 39].

Previous work have explored different approaches to understand drift. Concept drift is characterized based on probability distribution quantitatively [15, 21, 38], as well as on drift subject, frequency, transition, recurrence and magnitude [38]. Sources of drift are also discussed based on data distribution changes and decision boundary changes [21]. Despite this previous work, there is no existing taxonomy of data drift, particularly within the context of communications networks.

Another recent line of research focuses on explaining model drift; this problem has received attention in the context of malware detection, in particular. Statistical comparison of samples is used to identify model decay thresholds(i.e., decision boundary-based explanation) [18]. CADE uses contrastive learning to develop a distance-based explanation [39] to find the feature sets that have the largest distance changes towards the centroid in the latent space. Unfortunately, the explanation approaches are limited to classification problems and do not apply to prediction in general, such as the regression-based prediction problems we study in this paper. To the best of our knowledge, this paper is the first to explore a range of techniques to understand the features that contribute to model drift in regression-based prediction problems, incorporating a coordination of timeseries signal and dataset decomposition using domain specific knowledge and feature importance movements.

Machine learning models have been applied to many networking problems, including network anomaly detection [32, 33], network intrusion detection [10, 34], cognitive network management [5], and network forecasting [7, 25]. For instance, a framework is used to predict mobile network bandwidth in real time [25]. However, very few work has explicitly explored model drift along with it. Network traffic in both fixed and mobile contexts is inherently unstable over time; previous work has explored the nature of these changes. Mobile networks exhibit different patterns of activity during different time of day, day of week, and locations in the city of Rome, Italy [30]. Features such as latency can also drift over time: from network measurements over 17 months on more than 100 carriers, longitudinal shifts on the distribution of ping round trip time are observed [28]. Sommer and Paxson articulate that applying machine learning to anomaly detection is fundamentally hard in large-scale operational networks [35].

The COVID-19 pandemic has been an important setting for understanding how sudden drift can affect both network traffic patterns and resulting model accuracy. Such effects can significantly change the way users interact with the network, yielding significant shifts in traffic patterns and resulting network performance distributions. Significant variations in performance are observed across ISPs during COVID-19 lockdown [20]. Evident latency arises after the issuance of

stay-at-home orders after April. In Europe, traffic rised about 20% for broadband networks [13]. Blog posts [9, 27] showed a growth of 20.1% on the national downlink peak traffic in United States. UK mobile network operators reported drops of about 25% in downlink traffic volume [23]. Industry operators also self-reported network responses that causes the distribution of network capacity to drift [4, 8, 16, 24]. All of these factors can ultimately contribute to concept drift in predictive models, yet, to our knowledge, this paper is the first to explore the effects of these types of observed changes on model accuracy.

### 3 DATA DRIFT

In this section, we focus on data drift (i.e., changes in characteristics of inputs). We start with the description of our goal—forecasting target KPIs 180 days in the future—and a description of our dataset. We then demonstrate and taxonomize data drift using a single KPI, downlink volume at the eNodeB level, in the network.

#### 3.1 Context

**Problem.** Network forecasting is of utmost importance in capacity planning for cellular operators, which guides infrastructure configuration, management, and augmentation. In this paper, we focus on per-eNodeB<sup>1</sup> level Key Performance Indicator (KPI) forecasting to provide suggestions for machine learning models deployment, maintenance and operation in large cellular networks. Our goal is to use historical data to forecast KPIs of interest 180 days in the future. 180 days is chosen to reflect the provisioning window in practice. history of all KPIs in a training window, and our target is the value of a small subset of KPIs. In this study, we focus on the following target KPIs: downlink volume, throughput, peak active users, Radio Resource Control (RRC) establishment success, and S1 user plane external interface (S1-U) call drop rate.

**Dataset.** The analysis of model drift in this study is based on eNodeB-level LTE network measurements from a major wireless carrier in the United States. This dataset is from a metropolitan area and includes 224 daily KPIs for each eNodeB. KPIs are statistics collected in the network based on counters, measurements, and events generated by user equipment (UE) and radio access network (RAN) equipment. The KPIs are used to monitor and assess network performance, resource utilization, and user experience, to further support planning, optimization, and troubleshooting. Some of the most relevant KPIs are related to traffic (data volume, number of connected devices), data speed (throughput), retainability (drop calls and connections), and accessibility (access failure).

---

<sup>1</sup>Evolved NodeBs, or the “base station” in the LTE architecture.

These KPIs can be related to voice or data connections, and some of them have separate directional measurements: the downlink where the network is transmitting the data down to the UE’s and the uplink where the network is receiving data from the UEs.

The LTE RAN KPIs are mostly calculated using eNodeB counters. These counters are incremented based on network events that are generated or received by the eNodeB. Physical-layer RF environment conditions are also reported to the eNodeB. Some metrics are established to measure the network based on the LTE standard released by 3GPP [1], the body that determines cellular standards. These “raw” counters are kept by and can be collected from the eNodeB using external tools that are used to store data for longer periods of time, build formulas (which resemble KPIs but are generated from one or many underlying raw KPIs), and aggregate the same KPI for different granularities and levels. These metrics provide better visibility of the network performance and flexibility for analysis. A generic example of a formula to create a KPI is drop rate. This indicator is generated by dividing the number of abnormal releases by the number of successful established connections. As stated in Table 2, this dataset holds samples of 898 eNodeBs in a metro area and includes 737,577 daily KPI logs. Our dataset spans more than three years, from January 1st, 2018 to May 3rd, 2021.

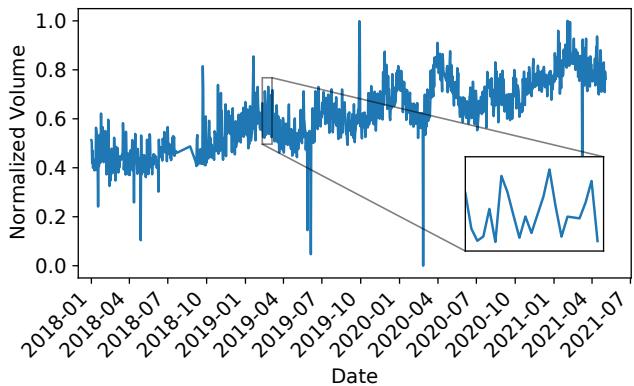
#### 3.2 Data Drift Taxonomy

Data drift is caused by distributional changes in features that are used as input in a machine learning pipeline. Such changes can be predictable or unpredictable; sudden or gradual; planned or unplanned; recurring or one-time. In this section, we provide a brief taxonomy of data drift using examples from our dataset. Figure 1 demonstrates long-term evolvements of downlink volume, with 3-week inset figures that provide a closer look at weekly patterns.

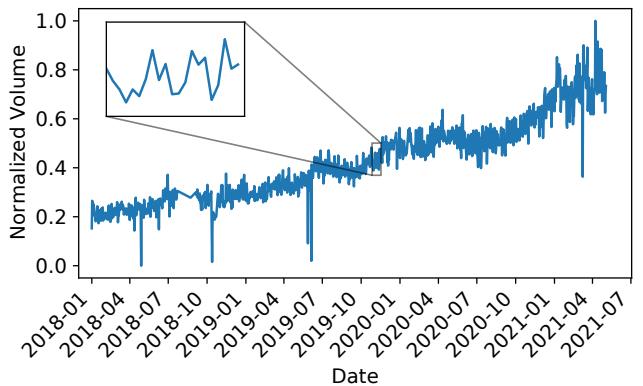
**Gradual drift.** Gradual drift occurs when the distribution of a variable changes over a period of time. Figure 1a shows an example of gradual drift using the data volume KPI for an eNodeB in our dataset. As shown, over the roughly 3.5 years in the plot, the volume increases for the eNodeB from 0.4 normalized data volume to a somewhat steady point around 0.8 in the latter half of 2020<sup>2</sup>. Zooming in, the 3-week inset reveals a periodic pattern three times, which indicates a weekly component. Gradual drift can be caused by a number of factors in real-world networks. For example, cellular carriers continually augment and improve their infrastructure. Upgraded infrastructure could then allow customers to send

---

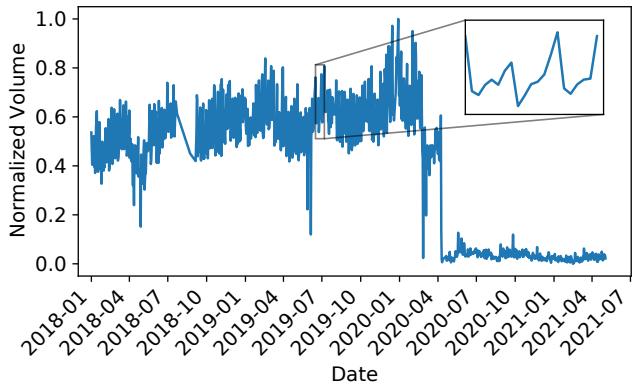
<sup>2</sup>Note that our examples of drift in operational networks may not perfectly resemble textbook definitions of data drift, as real-world network KPIs are inherently noisy.



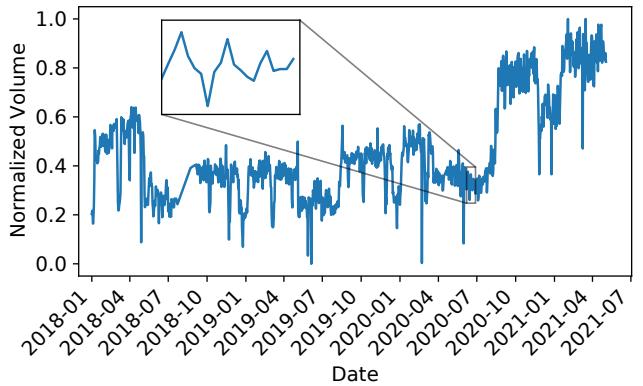
(a) *Gradual drift.*



(b) *Incremental drift.*



(c) *Sudden drift.*



(d) *Recurring context.*

**Figure 1:** Data drift taxonomy examples demonstrated by normalized downlink volume. Inset figures exhibit a 3-week view (all starting from Sunday) of normalized downlink volume for the box-selected period. Note that gradual and incremental drifts are different because incremental drift contains more intermediate distributions.

and receive more data on the network, causing the KPI to increase over time for a given eNodeB.

**Incremental drift.** Incremental drift is similar to gradual drift in that the variable shifts to a different distribution; however, incremental drift includes more distributions in the interim of a shift (*i.e.*, phase shifting from one steady state to another with a period of transition in-between). Figure 1d illustrates incremental drift for an eNodeB in our dataset, again using the data volume KPI. As the plot shows, volume on this eNodeB steadily increases over the observation period from roughly 0.2 normalized volume to 0.7. This incremental drift could be attributable to network changes such as the introduction of a new service plan and promotion. For example, carriers may add a plan that offers customers unlimited data. As users increasingly adopt such plans over metered plans, data volume increases for network eNodeBs.

Note that even if long-term evolution is incremental, we still observe weekly patterns on shorter timescales.

**Sudden drift.** Sudden drifts correspond to abrupt shifts in the dataset distribution. In the context of cellular networks, many events can be the cause for a possibly very noticeable, sudden drift. We can categorize these sources of sudden drift into *operator events*, (*e.g.*, network-wide upgrades), *scheduled events* (*e.g.*, a special event such as the Super Bowl), or *unexpected events* (*e.g.*, the COVID-19 pandemic or natural disasters).

Figure 1c shows a clear instance of sudden drift extracted from our dataset concerning total downlink volume for a given eNodeB. We observe that the normalized total drastically decreases starting in early 2020, then stabilizes to a consistently low average from March 2020 onwards (almost 60% less experience volume). These dates suggest that these sudden changes could be due to the events caused by the

COVID-19 pandemic where people’s mobility was heavily affected, and cellular networks with it [23].

In contrast, previous research into concept drift adaptation has primarily focused on how to minimize the reduction in accuracy by recovering using new data samples. Although this approach can help in some cases where the sudden changes are permanent (e.g., network updates), it can be counter-effective when applied to temporary and unpredictable shifts that cause one-off, ephemeral deviations in the data distribution (e.g., trends shown during the pandemic lockdowns).

**Recurring context.** Recurring context differs from previous categories as it does not refer to a permanent or persistent change in the dataset distribution. In contrast, recurring context identifies changes that occur periodically in the data. Such changes are strongly present in networking data as multiple factors can affect network performance and behavior. For example, users’ behavior and mobility patterns vary during the days of the week (weekdays vs weekend). Weather can also cyclically affect network performance; for example, rain and foliage can negatively affect the radio environment, and thus the transmission rates of a base station.

Figure 1d demonstrates the presence of recurring context of one eNodeB in our dataset. From the figure, we observe that summer months (*i.e.*, May to September) periodically show up to 50% lower data volumes for the base station, possibly a product of a lesser users’ presence and therefore usage of this particular eNodeB. These recurring patterns are heavily affected in 2020, a byproduct of the pandemic that led to lower volume demand in March, *i.e.*, the moment when stay-home regulations started. Weekly pattern exists during stable months such as February 2020.

In the literature [15, 21], recurring context has normally been tackled by the use of historical data matching the current data distribution. This approach, however, can introduce new challenges for networking data, which typically experiences multiple types of drift at once, such as the contemporaneous sudden and recurring context drifts from the previous example.

## 4 CONCEPT DRIFT

Concept drift captures the changes of relationships between model inputs and outputs. To gain insight into concept drift in the context of forecasting target KPIs in a cellular network, we train forecasting models using our dataset. We explore how models can be affected by concept drift by training a number of models from different categories of regression-based predictors (details are in Section 4.1). Further, we study how different training set sizes and training periods can affect model precision accuracy. In this section, we focus on the phenomena around concept drift by altering the above parameters. Based on these observations, we discuss the

best metrics for drift-aware model selection. We find that Normalized Root Mean Squared Error (NRMSE) over time is a good metric for understanding model drift, as it offers an equitable comparison across multiple KPIs over extended periods of time.

### 4.1 Experiment setup

As stated in Section 3.1, our task is a 180-day forecasting problem: we aim to predict from a given day the value of a small number of KPIs 180 days in the future. To explore the performance of different state-of-the-art techniques automatically, we use the AutoGluon [2] AutoML pipeline. AutoGluon is used to quickly prototype deep learning and machine learning algorithms on various existing frameworks: LightGBM, XGBoost, CatBoost, Scikit-learn, MXNet, FastAI. For all selected models, it enables automatic data processing, architecture search, hyperparameter tuning, and model selection and ensembling. We emphasize that our goal in this work is *not* to create and tune models for maximum accuracy, but rather to demonstrate and better understand concept drift in a real-world network.

As part of this study, we select four different families of models: gradient boosting algorithms like LightGBM, LightGBMLarge, LightGBMXT, CatBoost, and XGBoost; bagging algorithms such as Random Forest, and Extra Trees; distance-based algorithms like KNeighbors; and neural networks based techniques MXNet and FastAI. Although AutoGluon can fine-tune each model’s hyperparameters by hand, we rely on the default auto-selection pipeline with the goal of maintaining a fair comparison and to make training scalable and efficient. For example, we use the default neural network structure in FastAI [11]. For categorical variables, we use a 7-layer embedding with a Dropout that equals 0.1. For continuous variables, a BatchNorm is used. FastAI concatenates the results, followed by blocks of BatchNorm, Dropout, Linear and ReLU. Among the blocks, the first skips BatchNorm and Dropout, the last skips the ReLU. Finally, an activation layer using sigmoid is attached at the end.

We develop models for each selected target KPI: volume, throughput, call drop rate, and peak number of attached users. As the input of the training pipeline, we use the history of all categorical and numerical KPIs from all eNodeBs present in our dataset. On the other hand, while we target the forecasting of the target KPIs on an eNodeB-by-eNodeB basis, we generate a single model for the entire network, *i.e.*, we create a single model capable of forecasting values for each individual base station. We discard string-based features (*e.g.*, eNodeB ID) because it might generate undesired bias that could affect accuracy when predicting eNodeBs not present in the training set, *i.e.*, text representation might be encoded to representations which are not meaningful for base stations that are not present at all times in the dataset.

During our experiments, we use as input to models features from a given period of time and explore concept drift for this model over time, *i.e.*, training set time period size and forecasting length as parameters).

We test model effectiveness by computing distances between prediction and ground truth by date, specifically Root Mean Squared Error (RMSE). To better understand drift across different KPIs whose natural operating value ranges are drastically different (*e.g.*, call drop rates are scalars less than 1, while downlink volume scalars are often greater than 75,000), we normalize it by daily using maxmin and derive a Normalized Root Mean Squared Error (NRMSE). Further, we show the performance of regression by the coefficient of determination (*i.e.*,  $R^2$ ) in text. In practice, NRMSE scores under 10% and  $R^2$  over 90% indicate that the regression model has very good prediction power [26].

We also test other metrics that include mean absolute error, explained variance score, pearson correlation, mean squared error, and median absolute error. We omit these metrics for brevity. We observe that all phenomena we describe in the following section using NRMSE also hold for these metrics.

## 4.2 Concept drift behavior for different KPIs

Concept drift occurs when relationships between model inputs and outputs change. Here we explore whether we can observe drift for the KPI forecasting task and additionally whether the drift patterns from different KPIs are similar. Therefore, we experiment with different KPIs. We train all the models mentioned in Section 4.1 for each target KPI. To keep all other variables the same, the inputs of models are completely unchanged, but we alter the target KPI to predict. We use a 90-day window of historical data from all eNodeBs with an end date of July 1, 2018 for our training data. Then these models are tested on data subset split by dates. NRMSEs between the predictions and ground truths are calculated daily.

For simplicity, we present one representative method for each regression algorithm family: XGBoost for boosting, Extra Trees MSE for bagging, and KNeighborsDist for distance-based explanations. The neural network trained by default architecture using FastAI failed to converge to a reasonable NRMSE, so we exclude it in Figure 2.

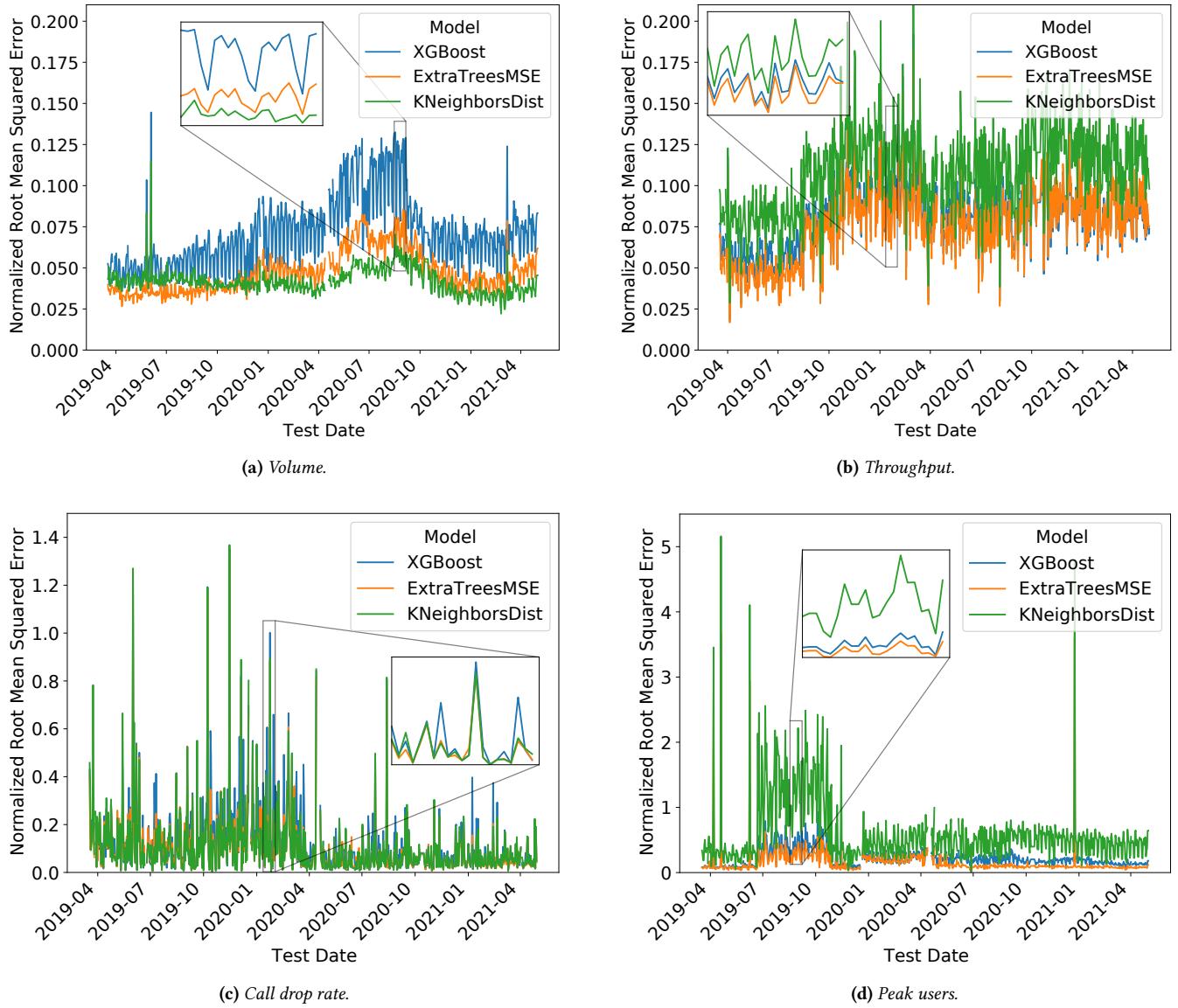
**Different KPIs exhibit different drift patterns.** Figure 2 presents the concept drift along with time for four classes of KPIs. Note that NRMSE for the model to forecast RRC establishment success exhibits similar drift patterns to downlink volume, so we only show only one plot for that pattern in main text. The plot of RRC establishment success can be found in Figure 11 in Appendix B. Overall, the drift patterns are quite unique for each class, and they vary in two aspects.

First, deviations in NRMSE occur at different periods of time. In Figure 2a, NRMSE of downlink volume experiences a sudden shift in April 2020 (corresponding to lockdowns due to COVID-19), and gradually drifts back to normal values in later months of 2020. Conversely, the throughput prediction model in Figure 2b witnesses a major increase in NMRSE around September 2019. Figure 2c exhibits higher values and larger fluctuations before COVID-19, but the model stabilizes and shows improved NMRSE values after April 2020. Moreover, short-lived, abrupt increases in error are much more frequent than for any other KPIs. For the prediction of peak users, Figure 2d demonstrates that July 2019 to November 2019 and January 2020 to April 2020 are harder to predict. In general, late 2019 and early-to-mid 2020 correlates with error drops and rises. This is mainly due to COVID-19 lockdowns as well as network upgrades.

Second, the high-frequency components have different patterns for different KPIs. By using signal processing techniques like STFT, no obvious weekly pattern is found on NRMSE of call drop rate. But if we look at the 3-week insets of Figure 2a and Figure 2d, 3 repetitions of similar signal patterns appear, which indicates a weekly pattern. However, the amplitude of such weekly drift is different. For example, the drift in volume has an amplitude of at most 0.025, but that of Peak users can be as large as greater than 1. Such performance could result from the magnitude of the original scalar values *i.e.*, volume has much larger average values than peak users. Therefore, a slight permutation of the peak users could result in a much larger drift.

**Different KPIs are most effectively predicted using different models.** As shown in Figure 2, we can find a model that performs relatively well for each target KPI. For all the target KPIs, there is at least one model with NRMSE less than 10% for at least half of a year. For example, even for peak users, the KPI that is most challenging to accurately predict, the average NRMSE from ExtraTreesMSE is 15.9% (8.9% for the first 100 days), and the  $R^2$  is as high as 82.2%. KNeighborsDist in volume, with a 4.2% average NRMSE and 93.7% for average  $R^2$ . The result using a neural network trained by FastAI (not plotted) is poor, however. It starts with an NRMSE score of 20.97% and ends with an average of about 4,282.05%. It also has a negative average  $R^2$  score, which indicates significantly poor fit to the data.

However, interestingly, models from different algorithm families have distinguishable effectiveness on different KPIs. For example, to our surprise, we found that the best model on one KPI might be the worst on another KPI. KNeighborsDist achieved the best performance over time on volume, with a relatively low NRMSE score around 5%. But it performs the worst when predicting throughput (11%) and peak users (mostly around 50% and at most 512%). ExtraTreesMSE, which has the best NRMSE over time for throughput and

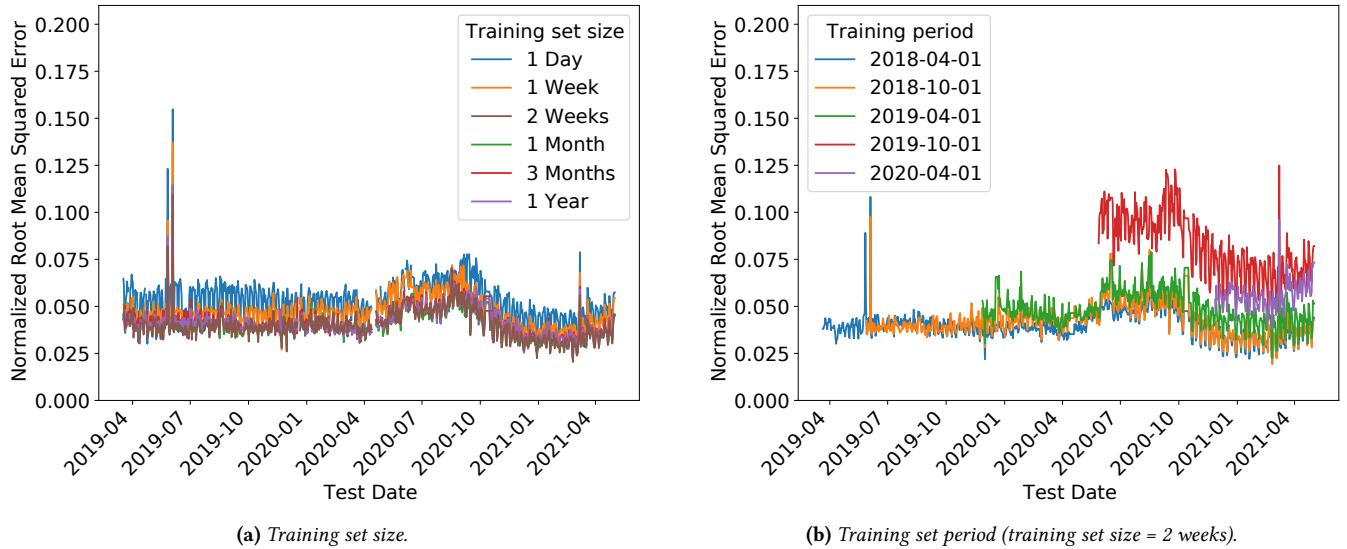


**Figure 2:** Drift of different models for KPIs of interests. Inset figures exhibit a 3-week view (all starting from Sunday) of NRMSE for the box-selected period.

peak users, is similar to or worse than that of KNeighbors-Dist. XGBoost does not significantly outperform the other two models, it performs nearly the best for throughput, call drop rate, and peak users. But it is the worst at predicting downlink volume.

One surprising effect is that model performance over time often runs counter to the observed performance from default metrics at training time. In a production environment, e.g., Amazon machine learning [3], and AutoGluon default pipeline, regression models are often chosen based on RMSE

during the validation phase. For example, in Figure 2a, if we were to pick the volume model using the default RMSE scores on the validation set, the best model is XGBoost (54,483.743); the worst model in this case is KNeighborsDist (91,842.678). However, this result is contradictory to the ranking when observing NRMSE over time. Thus, there may be opportunities to discover better metrics to select models in a drift-aware manner. We discuss this effect further in Section 4.4.



**Figure 3: Effects of training set size and training set period on concept drift.** In (a), model accuracy varies given different training set sizes, and the NRMSE mean of 2 weeks(4.07%) is even better than that of 1 year(4.17%). While in (b), the NRMSE mean of different training period are similar (~4.10%), except that of the training period 2019-10-1 has a much higher value (8.30%), possibly because that the model is trained on very abnormal prediction targets during COVID-19.

### 4.3 Concept drift behavior for individual KPIs

Our previous analysis of concept drift across multiple KPIs emphasizes the impact of the output of models. To understand how changes on the model itself could affect concept drift, we focus on a single KPI and explore differences in model family, training-set size and training period. In this section, we focus on a single KPI, downlink volume, as an example to provide insight.

**Individual KPI predictions drift consistently across different models.** As illustrated in Figure 2a, three different models are used to test the concept drift over time. For different models on the same KPI, the NRMSE of XGBoost, ExtraTreesMSE, and KNeighborsDist exhibit similar patterns. Although the amplitude of drift varies, the periods that include shifts in NRMSE are the same. KNeighborsDist is the same as XGBoost between April 2020 to October 2020, and then they stabilize together back to normal after COVID-19 lockdowns. This phenomenon can also be validated in other KPIs of Figure 2 as well: given a target KPI to forecast, all models on this task drift simultaneously. Even in the 3-week inset views, the signals drift similarly on the same date. This phenomenon also holds if we choose metrics like  $R^2$ , mean absolute error, explained variance score, pearson correlation, mean squared error, and median absolute error. Given our observations that KNeighborsDist is the best performing

model for downlink volume forecasting, we use it for the rest of the paper to simplify presentation.

**Consistent drift appears when trained using different training set sizes.** To investigate how parameters of a given model could impact concept drift, we test while varying the training set size. We vary the days of historical data from all eNodeBs with an end date of July 1st, 2018 and retrain models for each size training window. We use 1 day, 3 days, 1 week, 2 weeks, 1 month, 3 months, 6 months and a year. For clarity, we only show a subset of the above in Figure 3a.

As Figure 3a illustrates, all NRMSE values of different training set window sizes drifts similarly over time. While training set windows of 1 day and 1 week of data leads to a higher average NRMSE above 4.7%, the signal pattern remains the same: no matter what training set size, NRMSE experience a sudden increase after COVID-19 lockdown and gradually recovers after October 2020.

It is commonly believed that models trained on larger training sets (with more data variety) usually show better effectiveness [17, 36]. However, our results show that this conventional wisdom does not always hold true in practice: In Figure 3a, we see that the model effectiveness of two weeks performs *better* than that of 1 year. Its NRMSE remains at roughly 4.1% for the majority of the testing period. Such performance has practical impact on training regimes, as the size of the training set affects the cost to train a model. In Table 3, we record training time for different training set sizes and

algorithms. Using KNeighborsDist, training on two weeks is  $18x$  more efficient than 1 year<sup>3</sup>. Moreover, KNeighborDist is the most efficient model compared with all others, which requires only 0.107 seconds to complete training over 2 weeks of data. Using the same setting, ExtraTreesMSE takes 3.007 seconds ( $\sim 28x$ ), XGBoost takes 31.448 seconds ( $\sim 293x$ ), and NeuralNetFastAI takes 71.641 ( $\sim 669x$ ). On the other hand, the average NRMSE is as low as 4.07% for 2 weeks training set size, which outperforms other models. Given these results, we use 2 weeks for the training set window and KNeighbors for the rest of the paper if not otherwise specified. This optimizes the tradeoff between performance over time (*i.e.*, robustness) and efficiency.

**Consistent drift appears when trained on different periods.** Since we are testing the long-term effectiveness of our model, another concern is that whether the choice of the training period would affect the drift. In this experiment, we train on a number of different 14 day windows of historical data from all eNodeBs. In Figure 3b, the legend shows which 14 days are selected for training: 14 days of data before the date. For each training set, the model is tested daily in the future using a 180 day forecast.

Figure 3b also suggests that, no matter which training period, all NRMSE values drift simultaneously at similar timestamps. Again, although the red line (corresponding to a training period ending on October 1, 2019) shows significantly higher NRMSE scores (above 10% during COVID lockdown), NRMSEs from all training periods still exhibit a sudden bump after COVID-19 lockdown, and they gradually recover after October 2020.

Our experiment also suggests that models trained on more recent time periods do not necessarily result in better model performance. For example, the red line uses all the data from September 16, 2019 to October 1st, 2019 and the downlink volume from March 16, 2020 to April 1, 2020 as the forecasting target. This means that the model is trained on a rather abnormal period—at the beginning of the COVID-19 lockdowns. Doing so results in less accurate model predictions on the following test dates. This observation opens new possibilities for future research in exploring the best strategies for adapting to drift.

#### 4.4 Metrics for model selection

Based on our insights from Section 4.2 and Section 4.3, model selection in a drift-aware manner is different from more static models. Therefore, we propose NRMSE over time as a metric to select the best model. Our goal is to capture

<sup>3</sup>While the actual training times in our example are relatively small, our dataset consists of only a small portion of the mobile operator’s network. In practice, training on a nationwide network would result in significantly longer training times.

the long-term effectiveness of a model by looking into the mean and standard deviation of NRMSE. Low values of mean and standard deviation demonstrate the effectiveness and stabilities of a model. Rather than a single test (like RMSE) during training, NRMSE over time shows the evolutions on a whole time span and give a more appropriate overview.

Looking back at the example in Figure 2a, we can use NRMSE over time to assist with model selection. From default RMSE rankings, the best model is XGBoost (54,483.743); the worst model in this case is KNeighborsDist (91,842.678). Now we compute the mean and standard deviation of NRMSE time series. It is 6.38% and 1.72% for XGBoost, 4.63% and 1.17% for ExtraTreesMSE, and 4.27% and 0.77% for KNeighborsDist. Based on the long-term effectiveness and stability, KNeighborsDist is clearly the best model. Similar approaches can be applied to select the best training set size and training period as well.

## 5 WHAT CONTRIBUTES TO CONCEPT DRIFT?

The previous sections have illustrated that drift can arise when varying input, output, and models themselves. However, for many practical applications, it would be important to unearth more than the presence of drift, but also an understanding of its sources. In this section, we look to provide insight into concept drift from multiple perspectives. From a contributing factor perspective, we decompose the dataset and show the effects of morphology, traffic, and performance on drift. From a machine learning viewpoint, we explore the features that contribute the most to model performance by using permutation-based explanations. Lastly, from a signal processing point of view, we decompose frequency components to examine how inherent periodicity in the data can contribute to drift.

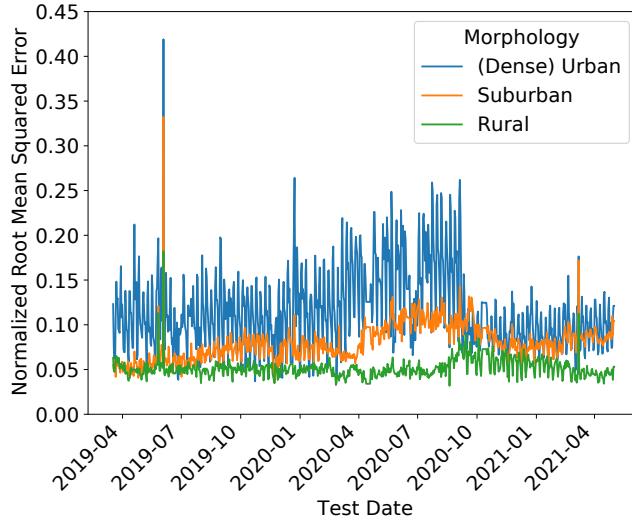
### 5.1 Effects of morphology, traffic, and performance on drift

To understand drift, we begin by exploring natural classes in the dataset, which leads to dataset decomposition. In this section, we classify the dataset based on morphology, traffic tiers, and performance tiers. We keep the model, the training set size and training period the same, *i.e.*, KNeighborsDist and 14 days of historical data with an end date of July 1, 2018. Moreover, we ensure that in each experiment, we have the same number of eNodeBs or number of logs for different classes. Then we retrain and test again based on these classes.

**5.1.1 Morphology.** We begin with the effects of morphology, where we label the eNodeBs based on the underlying population density of the regions that they serve. We classify eNodeBs as one of: dense urban, suburban, or rural. To

Set Size	Training Time / Average NRMSE						
	KNeighborsDist	ExtraTreesMSE	XGBoost	NeuralNetFastAI			
1 day	0.006	5.32%	0.746	5.56%	2.613	5.03%	61.579
1 week	0.042	4.73%	1.801	5.75%	10.365	7.05%	67.132
2 weeks	0.107	4.07%	3.007	4.88%	31.448	6.39%	71.641
1 months	0.271	4.09%	6.545	5.01%	55.808	6.77%	84.544
3 months	1.081	4.27%	30.651	4.68%	106.238	6.91%	127.942
1 year	1.924	4.17%	53.428	4.69%	142.971	6.68%	165.855

**Table 3:** Training time (in seconds) and average NRMSE for different algorithms using different training set sizes. The choice of 2 weeks using KNeighborsDist optimizes the tradeoff between training time and long-term performance.



**Figure 4:** Effects of morphology on concept drift. Lower population density leads to lower NRMSE sequences.

generate the classes, we look at a specified radius within a geographical location and establish different thresholds<sup>4</sup> and rank sites based on points of interest, population, and census blocks. Because the number of eNodeBs in urban and dense urban combined is 98, we constrain the measurements from suburban and rural to have the same number of eNodeBs.

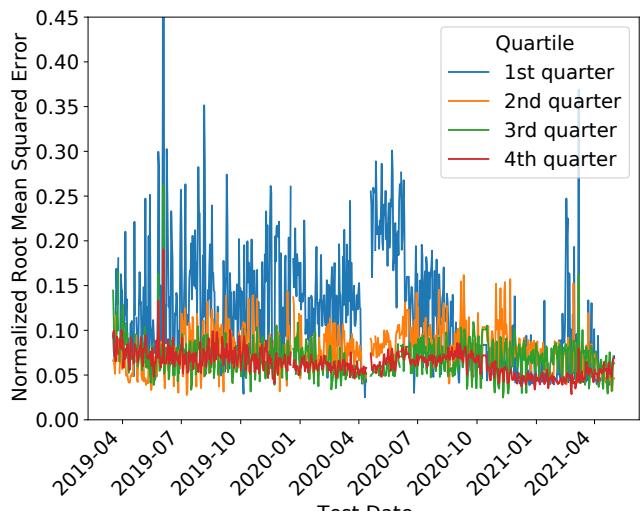
Figure 4 suggests that the higher the population density, the more abrupt the concept drift. First, higher population density leads to higher values of NRMSE on average. For example, before the COVID-19 lockdown period, dense urban areas have an average NRMSE of 11.4%, but the NMRSE for suburban areas is 7.9% and 5.2% for rural. This aligns with our observations that eNodeBs in higher population density areas serve more users and face greater mobility dynamism, which can lead to decreased prediction accuracy. Second, higher population densities correlate with more abrupt weekly fluctuations. The amplitude of NRMSE for

<sup>4</sup>Specific threshold values for different morphologies are proprietary information for the network that we studied.

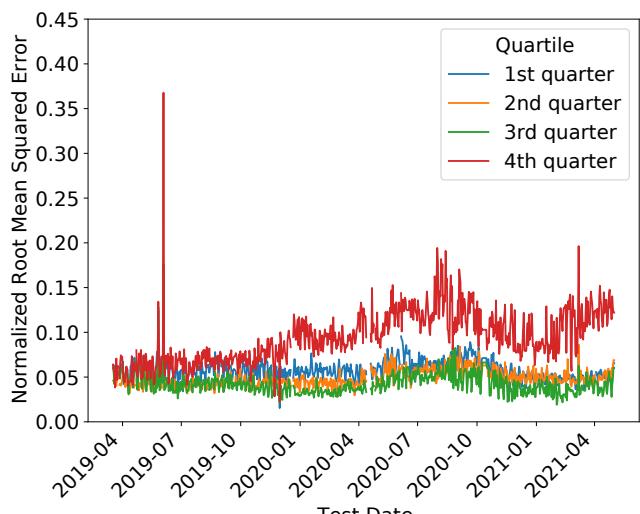
predicting urban eNodeBs is 5.54%, but it is 1.51% for suburban and 1.11% for rural eNodeBs. Urban areas have more extreme mobility patterns during weekdays and that contributes to higher weekly shifts. Third, higher population density regions tended to experience effects due to COVID-19 lockdowns earlier than other regions. As shown, urban eNodeBs are more sensitive to stay-at-home orders, possibly because of the changes in daily habits for workers and residents of the region. For urban eNodeBs, the actual drift occurs around March 2020. Conversely, suburban eNodeBs exhibit the shift after April 2020, and not until late September 2020 do the rural ones witness some drift. The distinctions based on morphology shed light on drift adaptation strategies. For instance, it could be argued from the network operator’s perspective that model retraining should be done more frequently for high-density areas, while low-density rural eNodeBs may be accurately predicted using infrequently trained models.

**5.1.2 Traffic.** Next, we classify the eNodeBs in the dataset based on traffic tiers. We split the dataset into four equal-size parts partitioned by 1st quartile, median, and 3rd quartile based on the downlink volume KPI. Then we retrain models and test by date. As shown in Figure 5a, data subsets with higher traffic history have better predictions. The average NMRSE is lower for higher volume eNodeBs compared to low-volume eNodeBs, and the weekly patterns show smaller fluctuations. This may be because that for a volume forecasting problem, if we decompose the dataset based on volume, even though the model performance on raw RMSE could be similar, when normalized the same amount of deviation would result in more concept drift. Together with Figure 10a in the Appendix, the NRMSE of under 5% volume is much higher than the over 95% class. This means that statistical outliers are more concentrated at the bottom tiers because the higher traffic tier is more stable.

**5.1.3 Performance.** Next, we again split the dataset into quartiles, but here we classify based on the throughput KPI.



(a) Traffic (Volume).



(b) Performance (Throughput).

**Figure 5:** Effects of traffic and performance tiers on concept drift. The lowest quarter in traffic and the highest quarter in performance exhibit the highest NRMSE.

In Figure 5b, again, we see that from 1st quartile to 3rd quartile, the average NRMSE is less and the weekly component has slightly smaller fluctuations. The 4th quartile, though, which is the group of eNodeBs with the highest throughput values, experiences the higher NMRSE values. Together with Figure 10b in the Appendix, the NRMSE of above 95% throughput is much higher than the below 5% class. This means that more outliers exist at the higher throughput tiers. Moreover, through analysis of the dataset we found that,

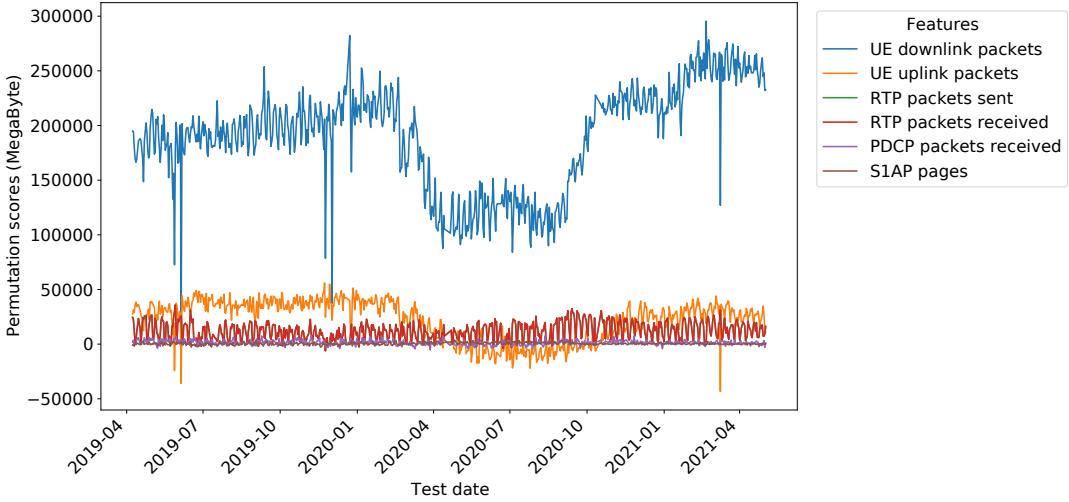
during COVID-19, high throughput values were rarer and therefore harder to predict.

## 5.2 Permutation-based feature importance drift

Next, we seek to understand the input features, and their relative importance, that drive model performance. To do so, we discover the most sensitive features through permutation. Using AutoGluon [2], we measure the performance degradation caused when the model predicts a permuted copy of the dataset, where the value of the feature has been randomly shuffled between rows. We subsequently compute the permutation score (*i.e.*, the distance between the original predicted value and the permuted prediction) of a feature. For example, a permutation score of 0.01 means that when a given feature is randomly shuffled, the prediction performance decreases by 0.01. Therefore, the higher the score for a feature, the more important it is regarding the effectiveness of the model.

We again use KNeighborsDist and 14 days of historical data with an end date of July 1st, 2018 for the model. We then apply the permutation-based approach to each data subset split by dates. Figure 6 demonstrates the top 6 features that most contribute to model performance using the volume KPI, meaning the permutation score related to the change in the number of MB predicted. As shown, the top three features dominate the majority of the predicted volume changes using this technique. Before the COVID-19 lockdown, permutations on User Equipment (*i.e.*UE) downlink packets can cause the predicted downlink volume to deviate over 200k MB. At the same time, the UE uplink packets have a permutation score of 48k MB on average. Then beginning in early 2020, effect related to COVID-19 impact the network, and these two features both experience significant drops. UE downlink packets drops to roughly 120k MB, while UE uplink packets even drops below 0, which indicates that the feature has begun to harm the prediction performance. After October 2020, the permutation score of UE downlink packets returns to its prior range and even increases, reaching up to 250k MB. Over the entire period, the third most important feature using the permutation technique is the Real-time Transport Protocol (*i.e.*RTP), which tends to maintain a relatively stable permutation score.

We find that the decreases of the top two features are negatively correlated with the rise of NRMSE. This phenomenon shows that some features were less important for prediction during COVID-19, as the network KPIs during the period were different than during times of normal network operation. The magnitude of permutation score decreases could be a good indication for drift sensitivity. It also implies that we can monitor concept drift by monitoring the evolution of feature importance.



**Figure 6:** Top 6 features that contribute to concept drift and their importance.

### 5.3 Drift components based on frequency decomposition

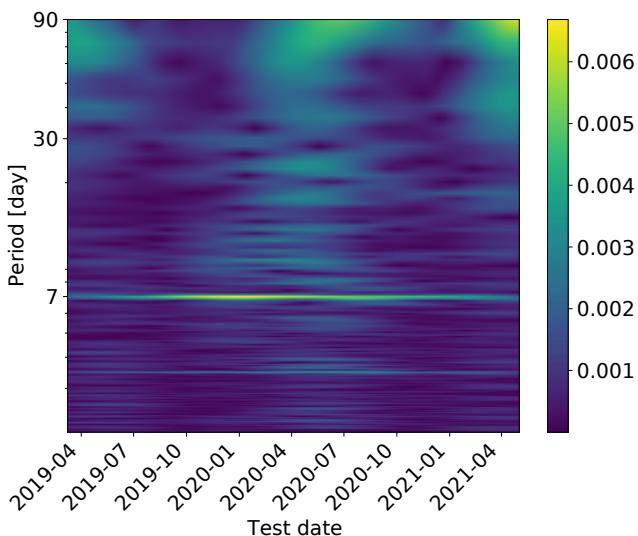
Given the temporal nature of the target KPIs in the network, we next look to apply signal processing techniques to further investigate drift components. In previous sections, we have seen strong weekly patterns in the NMRSE of KPIs in Figures 1 and 2. To formalize these patterns and identify the sources, we perform frequency-domain analysis on these signals. If we view the NRMSE time series output as a short-time signal that has a sampling rate of one day, then we can apply a Short Time Fourier Transform (STFT) to the time series. STFT can assist us in discovering major frequency components and then we can use a bandpass filter to yield a filtered signal with a specified frequency.

We take the output of XGBoost model shown in Figure 2a as an example to analyze. The phenomenon on other models and other KPIs are similar, and we omit them for brevity. Since this is a signal with relatively short duration—containing 765 date-dependant NRMSE scores, we use a sliding window of 365 days that slides one day for each sample to compute the Fourier Transform separately. Figure 7a shows the STFT of NRMSE time series output from XGBoost on the downlink volume KPI. We plot the y-axis in log-scale in order to show lower frequency values more clearly. On the x-axis, each test date represent the Fourier Transform executed on this date and the NRMSE in the next whole year, e.g., July 1st, 2019 actually contains all NRMSE from that date to July 1, 2020. Unsurprisingly, as shown in the figure, there is a very strong component on the period of 7 days in the STFT, illustrating a consistent weekly pattern. Note that there's also an apparent pattern around 3.5, which is due to the harmonic effect of the Fourier Transform. We also observe seasonal effects on the

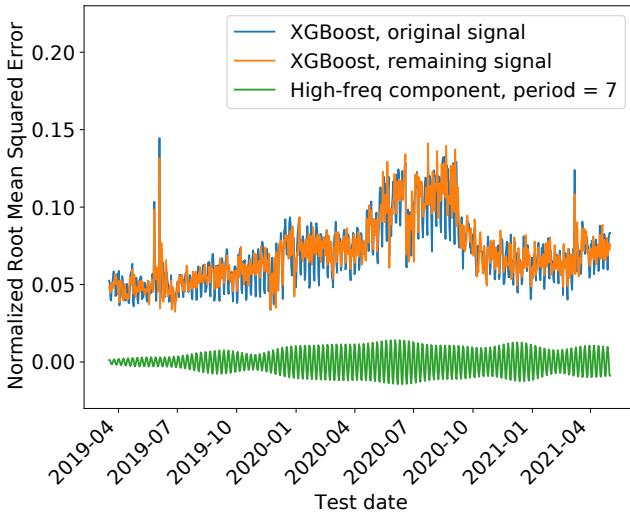
components, and we see a periodicity shift around 90 days. However, from July 2019 to April 2020 and October 2020 to January 2021, appears on the 90-day components. During COVID-19 lockdown, components of many different periodicity appear vertically in Figure 7a. We expect that these changes we attributable to a fundamental shift in signals during COVID-19, manifesting in abnormal drift.

**High-frequency drift.** As discussed, there is a strong 7-day periodic pattern in Figure 7a; this pervasive high-frequency component contributes to the drift on shorter timescales. Therefore, we seek to remove it from the signal using a Butterworth bandpass filter. In Figure 7b, we use signal decomposition to remove the weekly component and plot the original signal, the 7-day component, and the remaining signal. As shown, the amplitude of the weekly signal is relatively higher after COVID-19 lockdowns occur around April 2020. This means that a stronger fluctuation of prediction errors happens. Because this model is trained 15 months before COVID-19, it fails to capture more abrupt variance for days of the week during COVID. Over time, the amplitude of the removed signal gradually recovers, except for a slight increase around December 2020. This is possibly because mobility changes during COVID are more difficult to predict and vary more across different eNodeBs than during other time periods.

**Low-frequency drift.** Longer-term evolution in drift are evident in lower frequencies; these components can sometimes be attributed to seasonal (*i.e.*, weather-related) patterns. Figure 7a shows a clear low-frequency pattern around the periodicity of 90 days (roughly a season). The periodicity is discontinued between July 2019 to April 2020 and October 2020 to January 2021, though. In general, seasonality appears



(a) STFT Periodogram, whose y-axis is in log scale.



(b) Signal decomposition.

**Figure 7:** Signal characteristics on concept drift of XGBoost model output targeted to downlink volume. (b) shows the decomposed high-frequency component observed in (a).

to subside during winter periods of each year, which might be due to user behaviors such as the holiday season beginning around Thanksgiving to the new year being more irregular compared with other parts of the year. The low-frequency signal is especially strong during COVID. We believe this could partly be due to people remaining at home, which resulted in more stability in the signals.

**Exogenous shock.** Looking at the remaining signal in Figure 7b, we observe a clear indication of an exogenous shock during COVID-19 resulting in sudden drift. The NRMSE

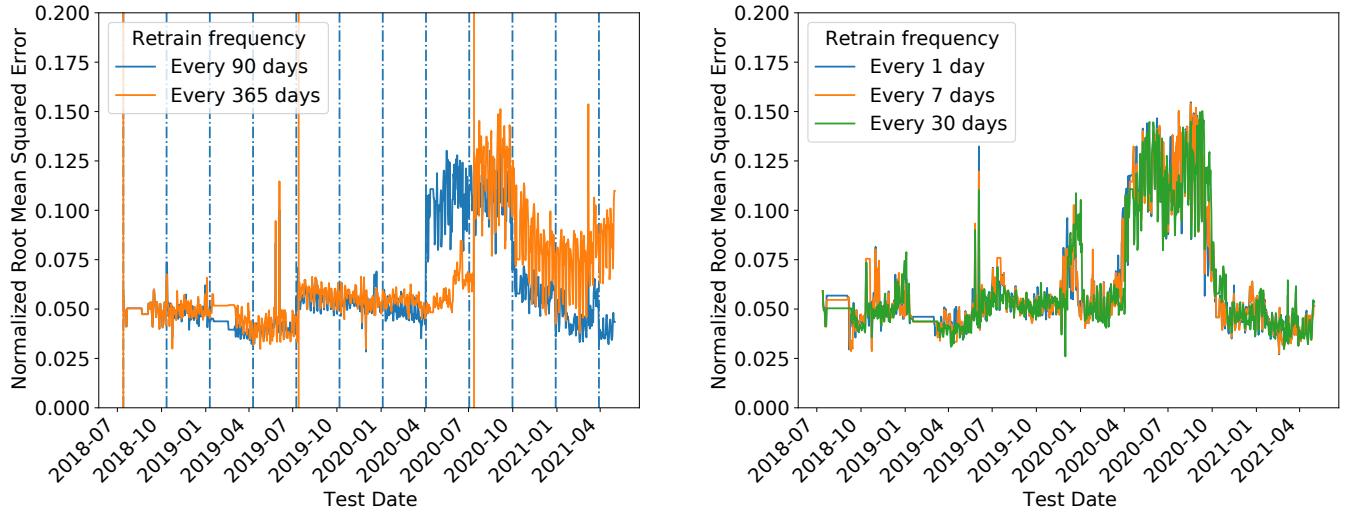
drifts from 7% to 12% during that period. This drift corresponds with a range of components, clearly visible as a vertical band in Figure 7a at the same time. Stronger components from 7 to 90 days indicate severe environment, user behavior, or network changes. Unexpected drift often occurs during disasters and other exogenous shocks [20], or potentially as a result of a network-wide upgrade (e.g., a wide-scale transition of users from 4G LTE to 5G could result in changes of both network and user behaviors).

## 6 DRIFT MITIGATION IN PRACTICE

In operational networks, a common intuition used to adapt to model drift is to simply retrain models regularly. Retraining using the most recent data is often considered an effective way to deal with concept drift. Many existing solutions [19, 37] adopt this approach. To understand whether this approach is indeed effective, we conduct an experiment to retrain models regularly using different retraining approaches, altering the frequency of retraining. As before, we use NRMSE to characterize the performance of models. Each model uses KNeighborsDist and a training set of 14 days. Given a retrain frequency  $N$ , a single model is used to derive the NRMSE for  $N$  days using 14 days of data, the model is then retrained iteratively every  $N$  days.

**Increasing retrain frequency does not mitigate drift.** To our surprise, we find that naïvely retraining regularly is an ineffective way to adapt to model drift. Figure 8 demonstrates the NRMSE resulting from this intuitive training strategy; vertical lines in Figure 8a indicate the points in time where the model was retrained. These two figures show that, no matter how often the retraining frequency is, the NRMSE during COVID-19 is as high as 12.5%, which clearly indicates the existence of concept drift. Furthermore, from Figure 8b we can tell that even if models are retrained at an unrealistic frequency of every day, concept drift occurs. However, as we see in Figure 8a, the model that is infrequently trained does not experience a significant increase in NMRSE until the retraining date. The model where the retrain frequency equals 365 days before July 2020, as shown by the orange line, shows a much lower NRMSE (6%) compared to the NRMSE (11.2%) of a model using a 90-day retrain frequency. This result aligns with what we observed in Section 4.3. Models trained on the most recent data may actually be adversely affected by training on data from an abnormal period.

**Lower prediction distance leads to reduced drift.** The reason behind the ineffectiveness shown in the previous subsection lies in the nature of a forecasting problem. The distribution of the input samples could be totally different from that of the prediction targets. It is the prediction distance (in days) in the forecast task that matters most in terms of the magnitude and width of drift. Figure 9 demonstrates



(a) Long term retraining. The vertical lines represent the timestamp we switch to a new model.

(b) Short term retraining.

**Figure 8:** Regular retraining using different retrain frequency. Note that the flat line means missing data during those periods.

the effects of prediction distance on drift. In Figure 9a, when the retraining frequency is every 7 days, higher prediction distance results in NMRSE values with higher magnitude and longer duration compared with lower prediction distances. For example, forecasting volume in 90 days have an average NRMSE of over 12.5% between July 2020 to October 2020. While the drift is at most 6.23% for 1-day prediction, 11.7% for 30-day prediction. Additionally, the drift days with high NRMSE scores are smaller, August to September 2020 for 1-day prediction and August to October 2020 for 30 days. Similar phenomena are seen in Figure 9b with a lower 91-day<sup>5</sup> retrain frequency.

Finally, by comparing the results with the same prediction distance in Figure 9a and Figure 9b, we still see that retraining with higher frequency does not necessarily result in lower drift. For example, the 1-day and 30-day prediction tasks show *higher* NMRSE values in September, 2020 when retraining every 7 days versus every 91 days. One important takeaway from this result is that network operators must choose their retraining regimes carefully, taking into account that concept drift exists however far into the future to forecast KPIs.

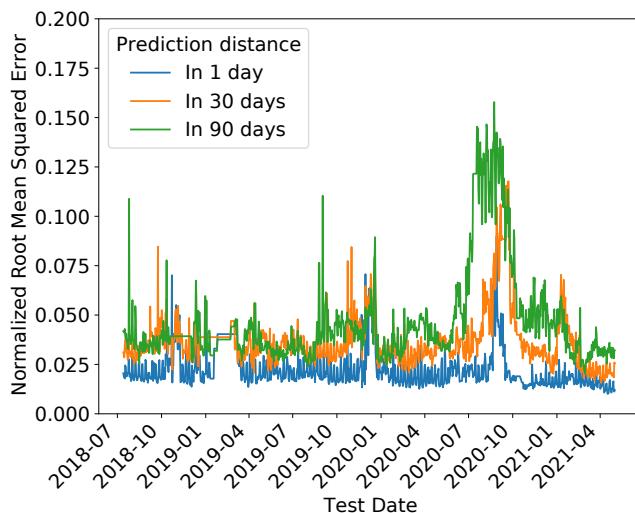
## 7 CONCLUSION AND FUTURE WORK

This paper characterized the sources of model drift in a large cellular network in the context of KPI forecasting. We analyzed and taxonomized data drift of various types, discovering and characterizing four different types of model drift.

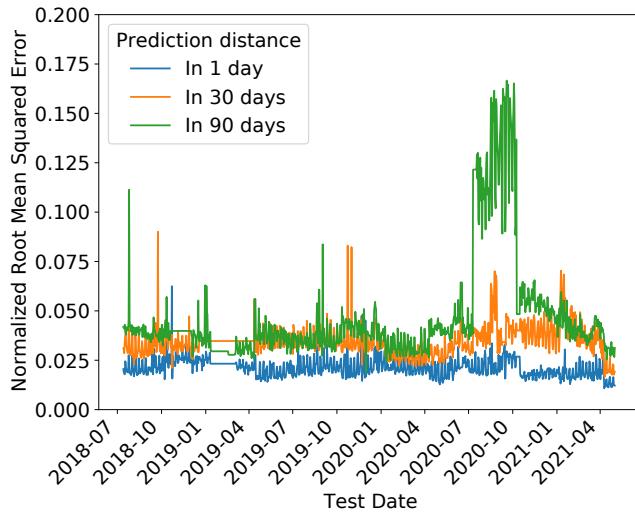
<sup>5</sup>We choose 91 days as it is a multiple of 7 and roughly a season.

We then explored how data drift can contribute to concept drift. We find that concept drift manifests in different ways for different KPIs, but for any given single KPI, the characteristics of drift are consistent, regardless of the choice of model, training set size, and training period. Building on our understanding of how concept drift manifests, we then seek to better understand whether specific features disproportionately contribute to concept drift. Using signal decomposition, dataset decomposition, and permutation-based importance explanations, we can determine the sources of the concept drift when predicting KPIs such as downlink volume forecast. Both high and low-frequency drift exhibit seasonal and weekly patterns (recurring context drift), while disasters and updates can cause shocks (sudden drift). Various aspects of morphology and traffic patterns also tend to coincide with model drift: We find that high population density, low traffic volume, and higher network throughput or speed also correspond to more concept drift. We also find that UE downlink data packets, UE uplink data packets, and total RTP received packets are the top three features that drift in ways that affect model performance over time. Our analysis also shows how sudden exogenous events, such as the COVID-19 lockdowns, can contribute to sudden drift as a result of changes in both user behavior (i.e., reduced mobility) and changes in traffic patterns (i.e., more working from home).

**Drift detection.** The drift detection problem in our context is challenging and unique for two reasons. First, instead of a classification problem well explored in the literature, especially in the context of malware detection [6, 18, 39], our



(a) Retrain frequency is every 7 days.



(b) Retrain frequency is every 91 days.

**Figure 9:** Regular retraining for tasks using different prediction distance. Prediction distance in  $N$  days means that this is a  $N$ -day forecasting task on downlink volume.

regression problem does not have a clear decision boundary that can help detection. The second is that a 180-day prediction task requires the target KPIs in half a year during the training phase. This imposes a significant challenge for forecasting: drift can occur when the distribution of the inputs does not change much, but that of the targets KPI changes. Thus, many existing techniques [15, 21] that focus only on dissimilarity of input samples and its significance will be ineffective for forecasting. In the network provisioning context, it is more important to examine changes to distributions

within input sample sets, as well as those between input samples and target samples.

**Drift adaptation.** While detection of drift is an important step, it is ultimately also important to adapt models in the face of the drift that is occurring. A drift adaptation scheme might incorporate one or more of the following strategies. One strategy might be to decompose components according to frequency. For high-frequency weekly patterns, the minor fluctuations can be removed by ensembling separate models for each day of the week, and subsequently training another model for the low-frequency seasonal patterns and shock. An ensemble of models trained on each day of the week could be effective as well. Our results also demonstrated that for some subsets of the data, the model can remain effective over time. For example, eNodeB in rural areas can be predicted using one model that does not require retraining; other examples include 2nd to 4th quarter from volume as well as 1st to 3rd quarter from throughput.

Thus, it is a good practice to train a model for classes with less drift. And then focus on the most drift-sensitive class to retrain. Another lesson from Figure 8 is to inspect on the input data distribution and avoid using abnormal data to train models. Thus, a third possible strategy to explore is to remove the features and data that do harm to the model performance when drift occurs. For example, in Section 5.2, we saw that the feature “UE uplink packets” exhibits negative permutation scores during COVID-19, which means that this feature makes the model less accurate in this phase. On the other hand, the tradeoff between long-term performance and drift mitigation effectiveness should be carefully examined.

**Drift explanation.** Another future direction is to develop better explanations for the factors that contribute to concept drift. Section 5.2 presented a permutation-based approach to help understand the features that best contribute to drift. Although this approach can help find the most drift-sensitive features by looking at its importance on the model performance, it nonetheless does not directly determine the features that contribute the most to the drift. For different features in the training set, randomly permuted features with other samples makes the comparisons unfair. Because the permutation might not have the same amount or portion of changes for each feature. Motivated by this observation and previous work on model drift in malware detection [39], a future direction could involve using a distance-based explanation that aims to find a feature set that moves a drifting sample towards the sample with the smallest NRMSE. In lieu of permutation, we could substitute the values of certain features with those from the lowest NRMSE sample. Based on this perturbation, one could observe the changes of distance in the latent space and find the most important features, which decrease the most distance.

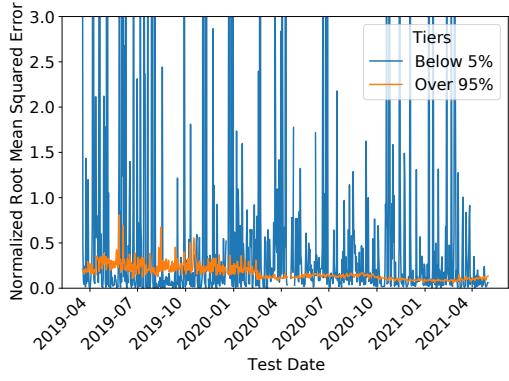
In summary, the findings in this paper indicate that concept drift in communications networks is an important, complex topic. Notably, a model that performs well at a given point in time, with a certain set of KPIs and a particular dataset may experience degraded performance over time. Operationalization of models thus needs to involve not only demonstrating that models perform well on offline traces in one-shot evaluations, but also that they can be maintained so that they continue to perform well over time. Future work in this area will be important both for the research community and for operations. When evaluating the performance of predictive models for networking tasks, the evaluation of new methods should take into account how the model performs over time, whenever possible. Already the results in this paper have demonstrated that the state-of-the-art approaches for mitigating model drift (e.g., frequent retraining) can be ineffective due to the contributions to model drift from recurring context. From an operations perspective, a deeper understanding of drift—especially detection, explanation, and mitigation—can help operators maintain more accurate models over time, and do so more efficiently. We briefly explore opportunities for future work in each of these areas.

## REFERENCES

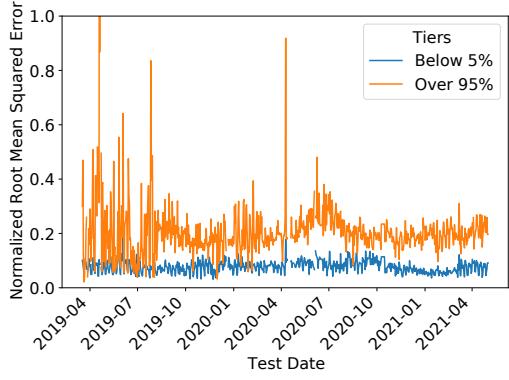
- [1] 3GPP. accessed July, 2021. *LTE standard, 3GPP TS 32.425 version 9.5.0 Release 9*. [https://www.etsi.org/deliver/etsi\\_ts/132400\\_132499/132425/09.05.00\\_60-ts\\_132425v090500p.pdf](https://www.etsi.org/deliver/etsi_ts/132400_132499/132425/09.05.00_60-ts_132425v090500p.pdf).
- [2] AutoGluon AI. accessed July, 2021. *AutoGluon: AutoML for Text, Image, and Tabular Data*. <https://auto.gluon.ai/stable/index.html>.
- [3] Amazon. accessed August, 2021. *Evaluating model accuracy: Regression - Amazon machine learning developer guide*. <https://docs.aws.amazon.com/machine-learning/latest/dg/regression.html>.
- [4] AT&T. accessed October, 2020. *COVID-19: Our Response*. <https://about.att.com/pages/COVID-19.html>.
- [5] Sara Ayoubi, Noura Limam, Mohammad A Salahuddin, Nashid Shahriar, Raouf Boutaba, Felipe Estrada-Solano, and Oscar M Caicedo. 2018. Machine learning for cognitive network management. *IEEE Communications Magazine* 56, 1 (2018), 158–165.
- [6] Federico Barbero, Feargus Pendlebury, Fabio Pierazzi, and Lorenzo Cavallaro. 2020. Transcending transcend: Revisiting malware classification with conformal evaluation. *arXiv preprint arXiv:2010.03856* (2020).
- [7] Sandeep Chinchali, Pan Hu, Tianshu Chu, Manu Sharma, Manu Bansal, Rakesh Misra, Marco Pavone, and Sachin Katti. 2018. Cellular network traffic scheduling with deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*.
- [8] Comcast. accessed October, 2020. *COVID-19 Network Update*. <https://corporate.comcast.com/covid-19/network/may-20-2020>.
- [9] CTIA. accessed October, 2020. *The Wireless Industry Responds to COVID-19: Network Performance*. <https://www.ctia.org/homepage/covid-19#network-performance>.
- [10] Bo Dong and Xue Wang. 2016. Comparison deep learning method to traditional methods using for network intrusion detection. In *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*. IEEE, 581–585.
- [11] fast.ai. accessed August, 2021. *FastAI v1.0, documentation*. <https://fastai1.fast.ai>.
- [12] Florentino Fdez-Riverola, Eva Lorenzo Iglesias, Fernando Díaz, José Ramón Méndez, and Juan M Corchado. 2007. Applying lazy learning algorithms to tackle concept drift in spam filtering. *Expert Systems with Applications* 33, 1 (2007), 36–48.
- [13] Anja Feldmann, Oliver Gasser, Franziska Lichtblau, Enric Pujol, Ingmar Poese, Christoph Dietzel, Daniel Wagner, Matthias Wichtlhuber, Juan Tapidor, Narsoo Vallina-Rodriguez, et al. 2020. The Lockdown Effect: Implications of the COVID-19 Pandemic on Internet Traffic. In *Internet Measurement Conference (IMC '20)*.
- [14] Futuriom. accessed August, 2021. *Verizon Applies Machine Learning to Operations*. <https://www.futuriom.com/articles/news/verizon-applies-machine-learning-to-operations/2018/08>.
- [15] João Gama, Indré Žliobaité, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)* 46, 4 (2014), 1–37.
- [16] Google. accessed October, 2020. *Keeping our network infrastructure strong amid COVID-19*. <https://blog.google/inside-google/infrastructure/keeping-our-network-infrastructure-strong-amid-covid-19/>.
- [17] Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems* 24, 2 (2009), 8–12.
- [18] Roberto Jordaney, Kumar Sharad, Santanu K Dash, Zhi Wang, Davide Papini, Ilya Nouretdinov, and Lorenzo Cavallaro. 2017. Transcend: Detecting concept drift in malware classification models. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*. 625–642.
- [19] Alex Kantchelian, Sadia Afroz, Ling Huang, Aylin Caliskan Islam, Brad Miller, Michael Carl Tschantz, Rachel Greenstadt, Anthony D Joseph, and JD Tygar. 2013. Approaches to adversarial drift. In *Proceedings of the 2013 ACM workshop on Artificial intelligence and security*. 99–110.
- [20] Shinan Liu, Paul Schmitt, Francesco Bronzino, and Nick Feamster. 2021. Characterizing Service Provider Response to the COVID-19 Pandemic in the United States. In *PAM 2021-Passive and Active Measurement Conference*.
- [21] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. 2018. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering* 31, 12 (2018), 2346–2363.
- [22] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. 2015. Recommender system application developments: a survey. *Decision Support Systems* 74 (2015), 12–32.
- [23] Andra Lutu, Diego Perino, Marcelo Bagnulo, Enrique Frias-Martinez, and Javad Khangosstar. 2020. A characterization of the covid-19 pandemic impact on a mobile network operator traffic. In *Proceedings of the ACM Internet Measurement Conference*. 19–33.
- [24] Martin McKey. accessed October, 2020. *Parts of a Whole: Effect of COVID-19 on US Internet Traffic*. <https://blogs.akamai.com/sitr/2020/04/parts-of-a-whole-effect-of-covid-19-on-us-internet-traffic.html>.
- [25] Lifan Mei, Runchen Hu, Houwei Cao, Yong Liu, Zifan Han, Feng Li, and Jin Li. 2020. Realtime mobile bandwidth prediction using LSTM neural network and Bayesian fusion. *Computer Networks* 182 (2020), 107515.
- [26] Robert Nau. 2014. Notes on linear regression analysis. *Fuqua School of Business, Duke University Retrieved from: https://people.duke.edu/~rnau/Notes\_on\_linear\_regression\_analysis--Robert\_Nau.pdf* (2014).
- [27] NCTA. accessed October, 2020. *COVID-19: How Cable's Internet Networks Are Performing: METRICS, TRENDS & OBSERVATIONS*. <https://www.ncta.com/COVIDdashboard>.
- [28] Ashkan Nikravesh, David R Choffnes, Ethan Katz-Bassett, Z Morley Mao, and Matt Welsh. 2014. Mobile network performance from user devices: A longitudinal, multidimensional analysis. In *International*

- Conference on Passive and Active Network Measurement*. Springer, 12–22.
- [29] Mykola Pechenizkiy, Jorn Bakker, I Žliobaitė, Andriy Ivannikov, and Tommi Kärkkäinen. 2010. Online mass flow prediction in CFB boilers with explicit detection of sudden concept drift. *ACM SIGKDD Explorations Newsletter* 11, 2 (2010), 109–116.
- [30] Jonathan Reades, Francesco Calabrese, and Carlo Ratti. 2009. Eigen-places: analysing cities using the space-time structure of the mobile phone network. *Environment and Planning B: Planning and Design* 36, 5 (2009), 824–836.
- [31] Jeffrey C Schlimmer and Richard H Granger. 1986. Incremental learning from noisy data. *Machine learning* 1, 3 (1986), 317–354.
- [32] Taeshik Shon, Yongdae Kim, Cheolwon Lee, and Jongsub Moon. 2005. A machine learning framework for network anomaly detection using SVM and GA. In *Proceedings from the sixth annual IEEE SMC information assurance workshop*. IEEE, 176–183.
- [33] Taeshik Shon and Jongsub Moon. 2007. A hybrid machine learning approach to network anomaly detection. *Information Sciences* 177, 18 (2007), 3799–3821.
- [34] Chris Sinclair, Lyn Pierce, and Sara Matzner. 1999. An application of machine learning to network intrusion detection. In *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)*. IEEE, 371–377.
- [35] Robin Sommer and Vern Paxson. 2010. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE symposium on security and privacy*. IEEE, 305–316.
- [36] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. 2017. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*. 843–852.
- [37] Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. 2011. Design and evaluation of a real-time url spam filtering service. In *2011 IEEE symposium on security and privacy*. IEEE, 447–462.
- [38] Geoffrey I Webb, Roy Hyde, Hong Cao, Hai Long Nguyen, and François Petitjean. 2016. Characterizing concept drift. *Data Mining and Knowledge Discovery* 30, 4 (2016), 964–994.
- [39] Limin Yang, Wenbo Guo, Qingying Hao, Arridhana Ciptadi, Ali Ahmazdadeh, Xinyu Xing, and Gang Wang. 2021. {CADE}: Detecting and Explaining Concept Drift Samples for Security Applications. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*.
- [40] Gan Zheng, Ioannis Krikidis, Christos Masouros, Stelios Timotheou, Dimitris-Alexandros Toumpakaris, and Zhiguo Ding. 2014. Rethinking the role of interference in wireless networks. *IEEE Communications Magazine* 52, 11 (2014), 152–158.
- [41] Indrė Žliobaitė. 2010. Adaptive training set formation. (2010).

## Appendix A EFFECTS OF TRAFFIC AND PERFORMANCE ON MODEL DRIFT



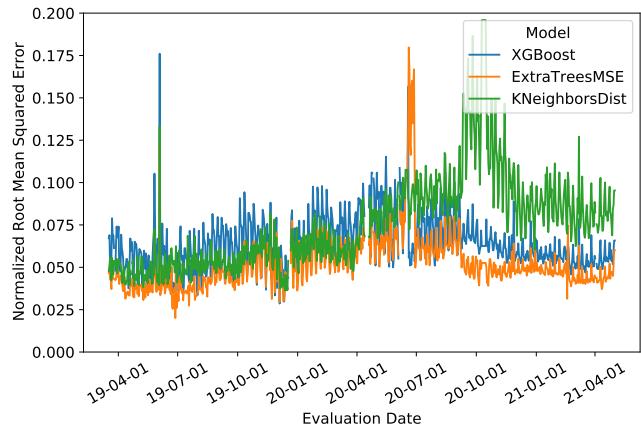
(a) Traffic (Volume).



(b) Performance (Throughput).

**Figure 10:** Effects of two major factors on concept drift in 5 and 95 percentile.

## Appendix B CONCEPT DRIFT BEHAVIOR



**Figure 11:** Drift of different models for RRC establishment success.