applied sciences

MDPI

Article

Outlier Detection with Explanations on Music Streaming Data: A Case Study with Danmark Music Group Ltd.

Jonas Herskind Sejr [1,*], Thorbjørn Christiansen [2], Nicolai Dvinge [2], Dan Hougesen [2] and Peter Schneider-Kamp [1] and Arthur Zimek [1]

[1] Department of Mathematics & Computer Science, University of Southern Denmark, 5230 Odense, Denmark; petersk@imada.sdu.dk (P.S.-K.); zimek@imada.sdu.dk (A.Z.)
[2] Danmark Music Group Ltd., Dartmouth TQ6 9BE, UK; bjorn@danmarkmusicgroup.com (T.C.); nicolai@danmarkmusicgroup.com (N.D.); dan@danmarkmusicgroup.com (D.H.)
* Correspondence: sejr@imada.sdu.dk; Tel.: +45-52-34-2918

Abstract: In the digital marketplaces, businesses can micro-monitor sales worldwide and in real-time. Due to the vast amounts of data, there is a pressing need for tools that automatically highlight changing trends and anomalous (outlier) behavior that is potentially interesting to users. In collaboration with Danmark Music Group Ltd. we developed an unsupervised system for this problem based on a predictive neural network. To make the method transparent to developers and users (musicians, music managers, etc.), the system delivers two levels of outlier explanations: the deviation from the model prediction, and the explanation of the model prediction. We demonstrate both types of outlier explanations to provide value to data scientists and developers during development, tuning, and evaluation. The quantitative and qualitative evaluation shows that the users find the identified trends and anomalies interesting and worth further investigation. Consequently, the system was integrated into the production system. We discuss the challenges in unsupervised parameter tuning and show that the system could be further improved with personalization and integration of additional information, unrelated to the raw outlier score.

Keywords: unsupervised; outlier explanation; lstm; forecasting

Citation: Herskind Sejr, J.; Christiansen, T.; Dvinge, N.; Hougesen, D.; Schneider-Kamp, P.; Zimek, A. Outlier Detection with Explanations on Music Streaming Data: A Case Study with Danmark Music Group Ltd. *Appl. Sci.* **2021**, *11*, 2270. https://doi.org/10.3390/app11052270

Academic Editor: Markus Goldstein

Received: 28 January 2021
Accepted: 2 March 2021
Published: 4 March 2021

1. Introduction

Today, businesses in the digital marketplaces can micro-monitor sales of products worldwide and in real-time. Market analysts are presented with large amounts of consumer data that contain valuable information about market trends. Having insights immediately when interesting events occur is essential and therefore tools are needed to detect changing trends and anomalies in real-time.

In collaboration with Danmark Music Group Ltd. (DMG) we have developed and deployed such a method that ranks and visualizes songs according to how much their streaming behavior deviates from the normal (i.e., expected) behavior. While extant work often revolves around recommending songs to end-users [1,2], our work, thus, focuses on recommending anomalous streaming behavior to backstage users such as studio executives, musicians and IT professionals.

We do not have access to purely normal data and therefore our method is unsupervised. The lack of supervision introduces challenges that we identified and solved. Both us, as developers of the method and the end-users need to understand underlying reasons for the identified outliers. Therefore our method supplies outlier explanations on two levels (Level 1 Explanation and Level 2 Explanation).

The model for expected behavior has in its core an artificial neural network with long short-term memory nodes (LSTM) [3] and the explanations are based on LIME [4]. The method is currently enabled for a subset of end-users (software developers, musicians,

and music managers) and the system has been evaluated on an offline dataset with 3end-users, both quantitatively and qualitatively. Along with the method and the application case study, we discuss the core challenges when applying unsupervised outlier detection and outlier explanation in real-life applications.

The novel contributions of our research are:

- A novel unsupervised LSTM based outlier method;
- Two levels of contextual outlier explanations;
- Discussion of challenges in a real-life application of unsupervised contextual outlier detection and explanation;
- Quantitative and qualitative evaluation with end-users;
- A discussion of context-dependent and personalized outlier detection based on empirical findings.

The remainder of the article is organized as follows. We discuss related work in Section 2. In Section 3.1, we discuss challenges with applying an unsupervised model for contextual outliers in general. In Section 3.2, we describe how we solved these challenges in a concrete application case study. We discuss the setup for quantitative evaluation in Section 3.3 and present the results from quantitative and qualitative evaluation in Section 4 before we finally discuss and conclude in Section 5.

2. Related Work

Detecting interesting developments and anomalies in time series data in some fields of research is treated under the terms "'trend detection" or "event detection" [5,6]. We will treat it as an "outlier detection" problem [7,8]. The close relationship between those areas has been discussed earlier [9].

Outlier detection in time series or streaming data covers a number of different types of outliers [10]. One type is outlier time series in a database of time series of equal length. Since a time series can be seen as a single point with as many dimensions as there are measurements, any traditional outlier detection method could be used [7]. By defining a similarity measure on the time series, density-based methods or cluster-based methods can find top outliers.

Point outliers or subsequence outliers [10] are other types of outliers in time series data. These outliers are sought within a single time series or multiple time series. Ignoring the context in which the points or subsequence outliers are sought, traditional outlier detection methods can also find these types of outliers, e.g., by defining an outlier score overall subsequences of similar length.

Another possibility is to define outlierness relative to the context, in which case they are called contextual outliers [7]. Contextual outlierness is defined relative to a context set of observations [11], e.g., a time window before or around the point. If the context is a window that includes the point itself, e.g., autoencoders can find anomalous points. Auto-encoders compress and decompress the data back to the original time series window and the reconstruction error is then used as an outlier score. For time series, LSTM auto-encoders capture timely dependencies [12]. The approach by Malhotra et al. [12] is a semi-supervised autoencoder, that assumes access to purely normal training data, but auto-encoders can also be trained fully unsupervised on a mixture of normal and abnormal data [13]. With this approach, the assumption is that due to anomalies being underrepresented, if restricted sufficiently, the model will only fit the normal points and the reconstruction error will be larger for abnormal points or subsequences.

Points or subsequences can also be analyzed relative to earlier measurements only. This type of outlier is rather intuitive and can be interpreted as how much the point or subsequence deviates from what was expected at a certain point in time. Novelty detection methods are of this type, since they define novelties based on prior measurements. Novelty detection methods typically do not consider temporal dependencies [14,15].

Looking at the temporal aspect, the obvious way to find outliers is to first predict the point or subsequence based on the past and then calculate a score that indicates how

well the real values match the expected. A plethora of methods exist for time series prediction (sometimes called forecasting) [16] and any of these methods could be used. Lately, LSTMs [3] have been shown to perform better than other methods [17] and LSTMs are becoming the preferred choice for time series outlier detection (e.g., [18]).

We apply a method similar to that of Malhotra et al. [18], but as we cannot assume access to purely normal data, our approach is unsupervised. This introduces profound challenges, identified in Section 3.1, and requires us to purposely underfit the model similar to how autoencoders work. Additionally, we deliver outlier explanations, deploy the method in a real-life application and evaluate with real users.

Explanation for supervised models is currently a hot topic [19–21]. Within outlier detection, a number of approaches have also been applied to explain the outlierness of found outliers (see, e.g., [22–28]), but none of these focus on contextual outliers or time-series outliers as we do.

3. Material and Methods

3.1. Background and Challenges

3.1.1. The Use Case

In the music streaming business, it is essential to know when streaming behavior changes for multiple reasons: Such information is an advantage in negotiation between stakeholders (artists, music managers, or record labels), it helps to focus marketing campaigns, and it may even help decide what the next song should sound like. DMG delivers a service with which artists and music companies upload songs to streaming services (Spotify, YouTube, iTunes, TikTok, etc.) and monitor streaming. The system aggregates streaming data from all of the streaming services and gives users a homogeneous interface that allows users to manually identify changing trends and anomalies. This task is hard and time-consuming, so in the ideal world, the system itself identifies and explains changing trends, failures in the system, etc., i.e., any anomaly that the user might find interesting. Our method addresses this use case.

In data science, finding these anomalies is called outlier detection. The typical definition of an outlier is "*an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism*" [29].

In the music streaming use case, the mechanism of "the other observations" is described by the distribution of the normal songs following expected trends. These distributions could, e.g., reflect that some songs are played more on the weekend or the dependency between number of streamings on consecutive days—the expected behavior. DMG is interested in surprises, i.e., outliers. Outliers differ from normal observations, e.g., when a song suddenly becomes more popular than expected or changes the pattern of when it is played. Such a change could be, e.g., the result of a concert or that the song is used in a YouTube video, in other words, changes to the mechanisms that generate the streamings.

Because outliers and inliers are relative we are not interested in a binary decision, but rather in a ranking of the observations on how normal or outlierish the objects are. The task is to find a model that defines the grade of normality (and outlierness) that ideally reflects how interesting the deviation is to the user.

3.1.2. Formalizing the Problem

Music streaming data (describing the total number of streamings per day) come themselves as a data stream. The data are time series data, and we want to analyse and act on the data immediately, when they arrive from the streaming services. We call the i'th time series of length m, T_i^m and the individual data point, e.g., at time j, $d_{i,j}$:

$$T_i^m = d_{i,1}, d_{i,2}, ..., d_{i,m} \tag{1}$$

In time series data every observation depends on previous observations based on their relation in time. Therefore new observations can be predicted to some extent based

on previous observations. We denote the window that we observe for outliers, the outlier window o with length l_o, and the window from which we predict the expected outlier window, we denote the context window c with length l_c. Starting from day j we thus have:

$$(c_{i,j}, o_{i,j}) = ((d_{i,j}, d_{i,j+1}, ..., d_{i,j+l_c}), (d_{i,j+l_c+1}, d_{i,j+l_c+2}, ..., d_{i,j+l_c+l_o})) \quad (2)$$

For example, if users monitor on a weekly basis, we want an outlier ranking of a 7 day window (i.e., $l_o = 7$). Assuming that there are monthly variations the context windows should capture at least roughly a month (i.e., $l_c = 30$).

The user wants to see surprises, i.e., music that has a different streaming count in the outlier window than would have been expected based on the context window. To predict the outlier window, we maintain a prediction model *pred* that takes a context window c and predicts the corresponding outlier window $\hat{o}$.

$$\hat{o} = pred(c) \quad (3)$$

We calculate the outlier score *os* as how much the real streaming counts deviate from the predicted streaming counts, according to some similarity function *sim*:

$$os = sim(o, \hat{o}) \quad (4)$$

Using machine learning we can automatically learn the prediction model from historic data and quickly updated the model, when data change significantly, e.g., when new streaming services are added.

3.1.3. Challenges

The unsupervised learning scenario incurs some fundamental challenges for outlier detection:

Challenge 1 (Outliers in the Training Data)**.** *Training data can contain unknown outliers that influence the prediction models.*

When outliers are defined relative to a model trained on past data, in the ideal situation the training dataset should not have any outlier subsequences. If we can guarantee that there are no outliers in the training data, we effectively have a semi-supervised scenario [30]. If the model learns, based on "dirty" training data, to predict outliers, the observed values will not deviate from the expected and we cannot identify them. This phenomenon is known as "masking" in the statistics literature [30]. A strategy to alleviate the problem is to remove the apparently most outlierish objects (e.g., those exhibiting the highest outlier scores) from the training data or limit the expressiveness of the model.

Challenge 2 (Outliers in the Context Window)**.** *The context window used for predicting the outlier windows will occasionally contain outliers.*

Even if Challenge 1 is somehow addressed, outliers can still be present in the context window and derail the predictions. Therefore we want the model to be robust against the influence of outliers in the context window. This relates to robustness in statistical methods [31]. A strategy that can alleviate this is to ensure that the model uses several context features in the predictions.

Challenge 3 (Lack of User Preference Data)**.** *We seek to detect outliers interesting to users, but in an unsupervised setting, there is no structured data on user preferences.*

Users find interesting what to them is unexpected, so our model should be complex enough to predict what the user predicts, and nothing else (due to Challenge 1). Without supervision in the form of labels, we cannot know if a model finds interesting

outliers or not and we cannot compare the quality of two models. Instead of labels, we need alternative ways of evaluating and getting feedback from users.

3.2. Application Case Study

In this section, we describe how the challenges identified in Section 3.1 are addressed in the case study with DMG (Figure 1).

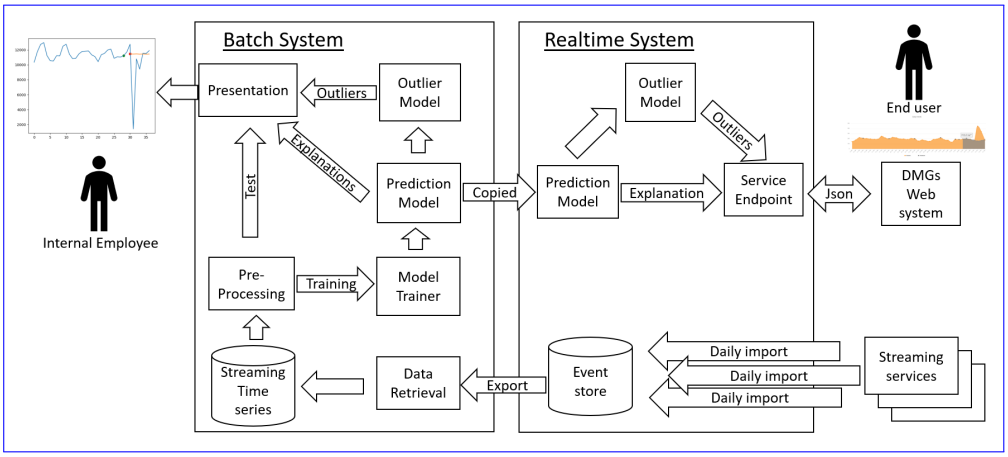


Figure 1. Events are daily imported from music streaming services such as Spotify or YouTube. When an update of the prediction models is required, events are imported and converted to streaming time series. In pre-processing data are filtered (series below a threshold may be removed) and split into a training set and a test set. These are used for the visual evaluation of outliers and explanations. Finally, the model is uploaded to the real-time system where predictions, explanation, and outlier analysis can be accessed by DMGs web front end.

3.2.1. Data Retrieval

In DMGs system, events from streaming services are daily imported and stored in a general format. Every event is a single streaming of a single song with annotated metadata (country, artist, etc.). These data are then regularly imported by the batch system. During the export, event data are converted into time-series data and grouped by song ID, resulting in a time series for each song with total number of streamings for each day.

3.2.2. Training and Test Data

Training and test samples are generated in preprocessing. With the test data, the user can evaluate the model before deploying it to the real-time system. For the results shown in the article the full window, from which training and test data are generated, spans from 1 May 2017 to 10 April 2020.

For the test samples to be useful in the evaluation of the model, the test set must not overlap with training samples. Therefore, the total time series window is split, so that test samples are generated based on the last $l_c + l_o + c_{test}$ days, where c_{test} is the number of test samples. Training samples are generated from a window starting at the beginning and ending at $c_{test} + l_o$ days from the last day. Thereby, there is no overlap between the context windows of the training samples and test samples.

$$D_{train} = \{(c_{i,j}, o_{i,j}) | \forall i, 0 < j < m - (c_{test} + l_o) - (l_c + l_o)\} \quad (5)$$

$$D_{test} = \{(c_{i,j}, o_{i,j}) | \forall i, m - (l_c + l_o + c_{test}) < j < m - (l_c + l_o)\} \quad (6)$$

Weekly correlation is typically strong and therefore we want an equal number of samples starting on each weekday. Because there could also be differences between the first part of a month and the last part of a month, we also want samples spread over a month, and therefore we chose $c_{test} = 28$.

Due to Challenge 1 we want to remove outlierish behavior from training data. From the perspective of DMG new songs in their catalog or songs removed from the catalogue are uninteresting outliers that should be removed to improve the model. Furthermore, DMG has less interest in songs with only very few streamings and the system will not be used to find outliers among these. We, therefore, filter songs with fewer streamings in the two periods c and o together than a threshold $t_{lim} = 10.000$. To ensure that there are streamings in both periods, time series with less than 10 percent of the threshold in either of the periods are also excluded.

With all songs and the interval used for evaluation, this process generates 252,378 training samples. Training with this many training samples is time consuming and takes around 10 h on the available machinery (CPU: Intel Exeon E5-2620v4, 2.1 GHz, GPU: nVidia GeForce GTX 1080 TI 11 GB GDDR5). We, therefore, select a random sample of 15,000 time series for training. Experiments with different sample sizes are described in Section 4.

3.2.3. The Prediction Model

Because of the way we will later explain predictions, we trained one model for each prediction in the outlier window and since we compare scores and explanations across songs, for each prediction in the outlier windows we trained one common model.

As discussed in Section 2 LSTMs have good performance on time series prediction tasks. Since neural networks also fit well with the large amounts of training data, we trained an LSTM for each day in the outlier window.

Songs have very different numbers of streamings so training data are normalized relative to the mean and standard deviation of the context windows (e.g., c_{train}). Likewise, when predicting a time series, it is normalized before the prediction is made and the result is denormalized before it is returned:

$$D_{train}^{norm} = \frac{D_{train} - mean(c_{train})}{std(c_{train})} \tag{7}$$

We want a network that will not learn the outliers (as perceived by the users, cf. Challenge 3) left in the training data, and at the same time, it should be good at predicting the inliers (Challenge 1). The amounts of data are an advantage and mean that it is possible to fit a complex normal behavior without fitting outlier behavior. The assumption here is that normal days are clustered in days of similar behavior, while outliers are either unique, within micro-clusters of similar behavior, or simply unpredictable. We have kept the neural network simple and added a dropout layer to regularize and increase robustness (Challenge 2).

The final neural network (Figure 2) was selected iteratively as the network with the best outlier ranking and outlier explanations, judged by the users. The model is optimized to minimizing the mean absolute error (*mae*). Training with *mae* instead of the usual mean squared error (*mse*) puts less effort in fitting outliers and will therefore contribute to solving Challenge 1. We train the model in 2000 epochs with a batch size of 32 using the Adam algorithm [32].

In Section 4, we describe experiments with different configurations of the neural network and the training process.

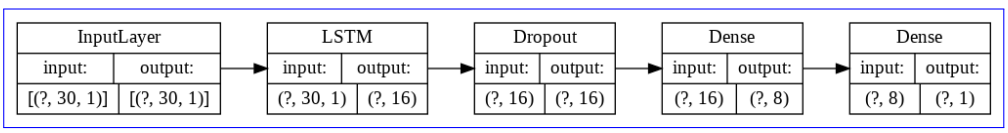


Figure 2. The final neural network evaluated and deployed to DMGs system. 7 prediction models are trained; one for each day in the outlier window.

3.2.4. Prediction-Based Outlier Scores

From the prediction of the outlier windows, we calculate outlier scores that express how much the real values deviate from the expected (i.e., the predicted) values. The outlier

score is calculated as the mean absolute error between the normalized predicted outlier week and the normalized real observations, where normalization is relative to mean and standard deviation of the context windows. The similarity function sim_{mae} for two vectors of length d is:

$$sim_{mae}(x, y) = \frac{1}{d} \sum_{n=1}^{d} |x_i - y_i| \quad (8)$$

We use *mae* because we expected users to be more interested in changing trends than single day outliers. With *mae* large deviations are not punished as hard as with *mse*. The final outlier score is therefore:

$$os = sim_{mae}(norm_c(o), pred_{lstm}(norm_c(c))) \quad (9)$$

where $norm_c$ is normalization relative to c and $pred_{lstm}$ is trained on D_{train}^{norm}.

3.2.5. Outlier Score Customization

To DMG the outlier score itself is not enough to prioritize the importance of the outlier. DMG also wants the total number of streamings to influence the outlier ranking. We did not find a good model for calculating an outlier score as an aggregate of outlierness and total number of streamings. Instead, the service available to developers in DMG, delivers outlier scores that non-scientist developers use for the ranking, before requesting predictions and explanations. In this way, developers can change the weighting of total streamings and raw outlierness without changing the outlier detection algorithm (addressing Challenge 3).

3.2.6. Level 1 Explanation

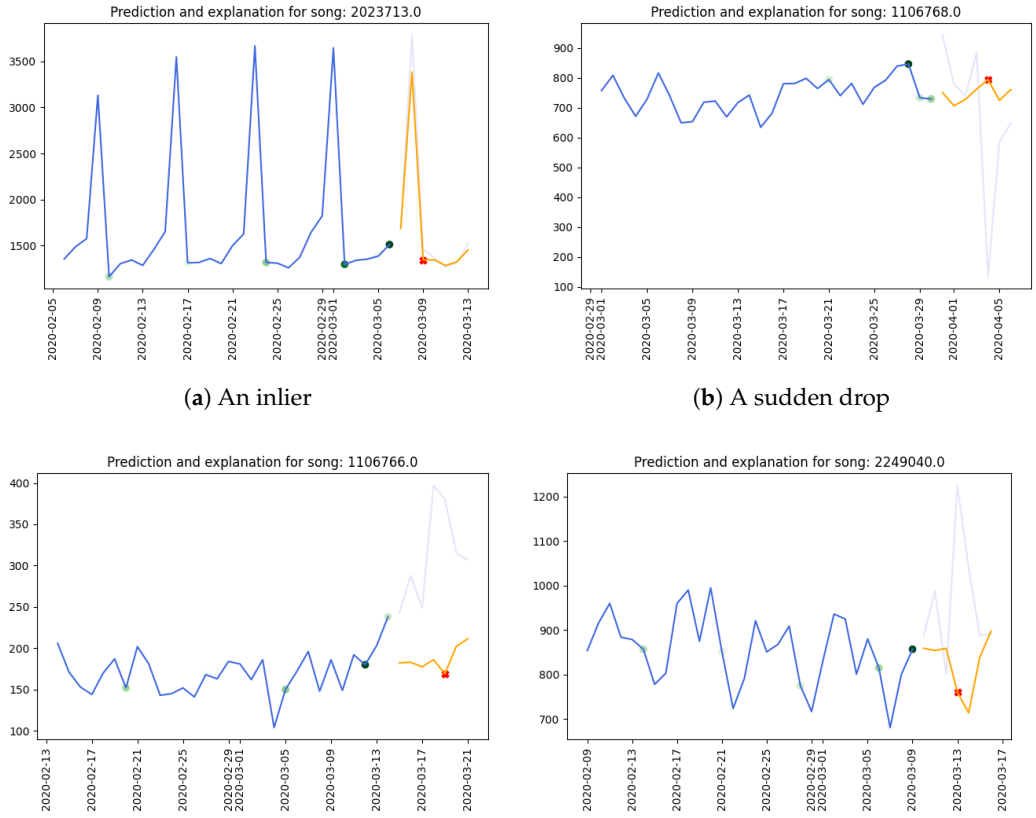
Using *mae* over *mse* contributes to the interpretability of the outlier score. With *mae* the outlier score is proportional to the area between the predicted and the real values (see Figure 3). This simple white box explanation, explains feature by feature how the observed values deviate from the expected. This is the typical way to explain non-contextual outliers [22–28].

3.2.7. Level 2 Explanation

Because we deal with contextual outliers the expected value depends on the contextual variables (the context window). Therefore we go one step deeper and explain the expected value highlighting the most influential contextual features. We explain the expected values using LIME [4], which is based on a local linear model. A set of local points are selected and LIME fits a linear model. By using a version of LASSO [33], the algorithm ensures the most influential features are selected.

3.2.8. Presentation

The Level 1 Explanation is presented by plotting the predicted values together with the actual values. For presenting the Level 2 Explanation we chose the most influential contextual feature together with contextual features with an explanatory weight above 30% of the maximum weight. In the presentation used for the offline evaluation (Figure 3), one plot is generated for each prediction. Selected explanatory features are colored according to their weight.



(**a**) An inlier (**b**) A sudden drop

(**c**) Song 1 (Band 1) (**d**) Song 2 (Band 2)

Figure 3. Static visualization for evaluation of 4 samples: One inlier Figure 3a and 3 songs with high outlier scores. The blue line (the context windows) and the gray line (the outlier windows) show the actual observations. The yellow line shows the prediction of the outlier window (Level 1 Explanation). In each visualization, a prediction is selected and marked with a red cross. The green markings within the context window show the most important features in the prediction and outlier score (Level 2 Explanation). The color indicates the importance.

3.3. Evaluation Setup

After tuning the system with user feedback we generated a single labeled dataset for a quantitative evaluation. The system was evaluated in a workshop with three users from DMG: A manager who is also a musician with songs published through DMG's system, a manager with programming experience, and a developer part of the team responsible for maintaining the system. Apart from representing multiple roles as users of the system, these users also have many years of experience in the business and are good proxies for other types of users.

To get a dataset on which we can compare different configurations we selected top 20 outliers from each of the executed method configurations. To this selection, we added 10 normal objects with low outlier scores. The users scored each of the samples from 1 to 5 reflecting how interesting they found the outlier, i.e., how much it deviates from their expectations. Even though the users had very different perspectives, we wanted a common grade for the outliers. Planning poker (Planning poker was original described in a white paper by James Grenning in 2002. The paper can be found at https://wingman-sw.com/papers/PlanningPoker-v1.1.pdf, accessed on 28 January 2021) is a quick method to find a common estimate, so we used a similar approach: Each user has 5 cards with grades from 1 to 5. On the count of three, all users show their grades. If they agree, the grade is written down and if not, the users with the top grade and bottom grade present their reasons, and the time series is reevaluated until the group agrees. The users quickly got a common

agreement on how they would evaluate the time series and they evaluated all samples within 2 h. To understand what is behind these grades, we asked the users to analyze the root causes of a sample of three outliers with grades above 3. Finally, we discussed with the users, the outlier explanations for the sample.

4. Results

4.1. Quantitative Results

The users are primarily interested in top-ranked outliers so we defined the quality measure *atg* as the average user grade of top 20 outliers for evaluation and comparison of the configurations we have executed (Figure 4). Results show that the deployed configuration has an average *atg* of 4.55 for the top 20 outliers (note that maximum $atg = 5$). The 10 inliers embedded into the evaluation data were graded with grade 1.

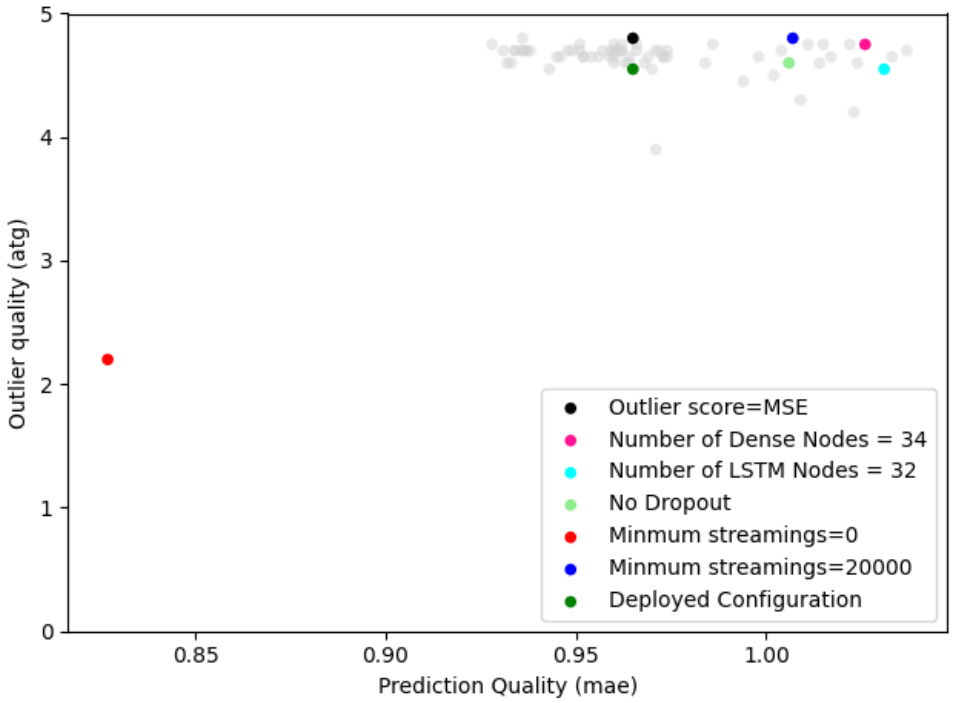


Figure 4. The deployed configuration has 16 LSTM nodes, 8 Dense nodes, a 0.5 dropout layer, 15,000 training samples, 2000 epochs, minimum streams in sample is 10,000, and the outlier calculation used is mean absolute error. We have plotted all the experiments we have executed to reveal the relation between precision (*mae*) and outlier quality (*atg*). Additionally we have labeled interesting changes to parameters.

4.1.1. Training Parameters

To separate model selection and training we made sure to have enough training samples and epochs to fit the model. Experiments showed that *mae* stabilized with around 15,000 samples and 2000 epochs.

4.1.2. Precision Versus Quality of Outlier Scores

We imagined outlier quality (*atg*) increasing with precision *mae* until some point, where the model starts predicting what the user perceives as outliers (Challenge 1). We learned that the relation between model expressiveness and outlier performance is more complicated than that (Figure 4), so precision cannot be used directly for model selection and parameter tuning.

4.1.3. Outlier Calculation

Before the user evaluation, we assumed single outliers in otherwise normal weeks to have low priority to the user. We later realized that the users in the workshop found single outlier days highly interesting: sudden drops can indicate an error in the system while positive spikes can be the result of concerts, advertisements, or similar. Changing the outlier score to use *mse*, *atg* increases to 4.80. According to the users, this could be different for other users who would find single-day anomalies less outlierish or unexpected. In that case, *mae* is a better choice.

4.1.4. Total Number of Streamings and Outliers Score

When ranking the outliers, the users were very clear that low volume songs are only interesting if outlierness is very strong. An increase of 50 streamings could for example come from a private event the musician attended, which is not the type of outliers interesting from a business perspective. Removing the threshold (Minimum Streamings = 0) decreases the outlier quality significantly (even though the predictive capabilities increase) while increasing the threshold (Minimum Streamings = 20,000) results in the highest outlier score we saw 4.80.

4.2. Qualitative Evaluation

4.2.1. The Value of Detecting Outliers

Even though finding the optimal parameters is not straightforward, all of the parameter choices we have experimented with, highlight time series that users find interesting. It is therefore also interesting what is behind the top outliers. An important pattern among top outliers was sudden drops (e.g., Figure 3b). The analysis shows that these drops are most likely caused by IT-related errors. Another observation was that many of the top outliers are from the same artist (Band 1) (Names anonymized.) (e.g., Figure 3c). The users agreed that these deviations most likely come from the same underlying event: one of the artist's songs has been added to a Spotify playlist which increased streaming of all of the artist's songs (e.g., through recommendations). With another song (Figure 3d) the users agreed that the increased interest in the song was due to a blog post by the artist published the day before.

4.2.2. The Value of Outlier Explanation

During the iterative process of selecting the model and parameters, both the researchers and the developers in DMG found both levels of explanations valuable. Having the predictions is valuable, e.g., in determining if the model fits outliers (Challenge 1). Having the explanation of the predictions helps to assess if the prediction model is robust (Challenge 2) and similar to the user's model of expectation (Challenge 3).

We observe that the learned model is not able to predict outliers and that the predictions are based on several previous observations. The strongest influence on the predictions comes from the latest observation and the same weekday in previous weeks, which aligns with the expected user intuition (Figure 3).

Having the predicted versus the observed inspired the end-users to discuss the validity of the outlier scores, but for end-users who are not used to think about learned models, model explanation (Level 2 Explanation) is hard to grasp and does not add much value. The end-users are more interested in causal explanations to the outlier than an explanation to the prediction.

5. Discussion

5.1. Evaluation and Adoption of The System

Our evaluation shows that the system is valuable to the users, with users grading the importance of the top 20 outliers to be on average 4.55, on a scale from 1 to 5. The quantitative evaluation was limited by the number of labeled time series, but the conclusion is strengthened by the clear coincidence between quantitative and qualitative

results described in Section 4.2.1. Because of the positive evaluation, the system is being further tested as part of DMGs production system and we are continually getting feedback with interesting findings based on top outliers.

In the production system, users request songs with the most surprising trends during the last week and for each top 3 outlier song, the user is presented with the trend together with the expected behavior (level 1 explanation) (Figure 5).

Even though end-users found it hard to grasp level 2 explanation, DMG has decided to keep it in the presentation, but only present the explanation when the users hover the expected values.

When the users hover a prediction, the most influential contextual features are highlighted with dots (level 2 explanation). Instead of coloring the dots according to feature importance, the most important explanatory features are all highlighted with black dots, reducing the amount of information presented to the user.

The current system is designed to find outlier songs, but—without changes to the outlier detection method–the system can also be used to identify outlier artists, outlier countries, etc., which will be a natural extension in the future. DMG will conduct experiments with different sources of outlierness.

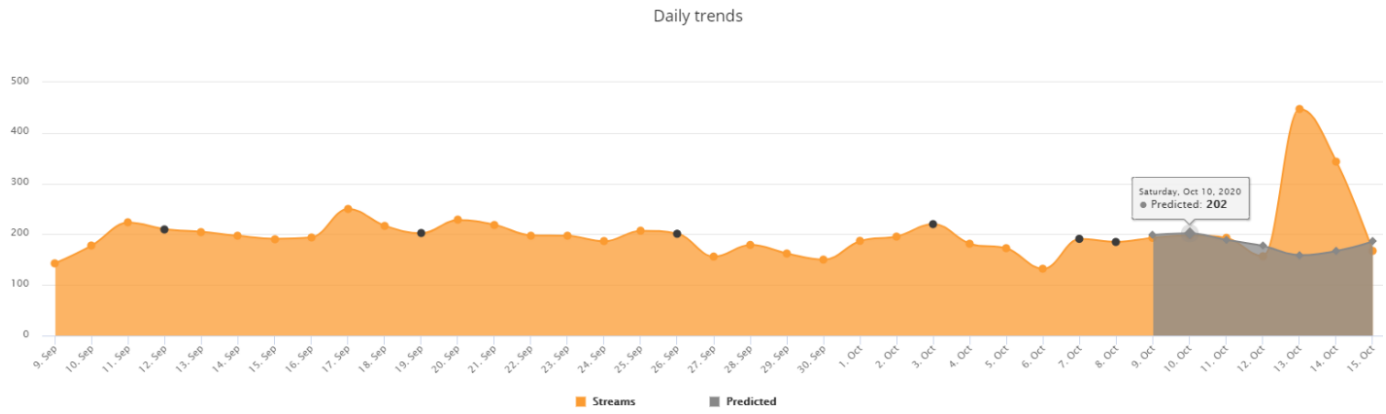


Figure 5. The presentation of outliers and explanations in DMGs monitoring system: Expected versus observed streaming count (Level 1 Explanation) are presented for each outlier and when the users hover the individual predictions the explanation of the predicted value is highlighted (Level 2 Explanation).

5.2. Finding the Best Parameters

The currently deployed configuration of the method was found through iterating with developers and end-users in DMG. During the iterations, we found both levels of explanations valuable. We have described finer parameter tuning in Section 4, but explanations were already valuable before the number of layers, the types of layers, normalization, etc. were fixed.

Often the iterative process of finding a model and tuning parameters in unsupervised settings is ignored, but this way a large part of the value gained from outlier explanations is omitted. Even with outlier explanations, we saw that the iterative process was both time consuming and limited and we found no simple pattern between the precision of the prediction model and outlier quality.

5.3. The Complexity of Outlierness

5.3.1. Personalization

In our use case, the only way to meaningfully define outlierness is relative to the user's perception of normality. The different perceptions and prioritization of single-day drops in streaming, exemplifies the context (in this case the user) dependency of outlierness. The ideal outlier detector, in this use case, is therefore personalized, but this can only be achieved with some type of supervision.

5.3.2. Non-Outlier Information

Changing the threshold for total number of streamings had the biggest effect on the outlier quality (Figure 4). With our model, total number of streamings does not influence the prioritization of outliers. However, to the users in DMG, medium-sized deviations are expected for songs with a small number of streamings and are much less interesting. An ideal outlier detector therefore must also take into account information not directly related to outlierness. DMG is currently experimenting with a balancing of outlier score and total number of streamings when prioritizing the outliers.

5.3.3. Outlier Grouping

When we identified multiple interesting outliers from the same artist the users found it valuable but added that they would prefer if the system only ranked one of these outliers, since they are related to the same underlying event. In other words, the users want a kind of alert correlation.

5.3.4. Outlier Explanations

Seeing the values expected by the model (Level 1 Explanation) increases users' trust in the model, but we found that explaining how the model found the expected value (Level 2 Explanation) was too abstract in the current presentation. Talking to the users they expressed a need for casual explanations more than a model explanation.

5.4. Conclusions and Future Work

In our case study, we have the advantage of being involved in the development process and therefore can evaluate the value of outlier explanation during development and implementation. On the other hand, we are limited by looking at a single system and three end-users.

This means that it is hard to generalize our results, but what we can conclude is, that it is possible to do prediction-based contextual outlier detection in a completely unsupervised setting. We also conclude that the two levels of contextual outlier explanation can be used during model selection and parameter tuning and that level 1 contextual outlier explanation can be valuable to the end-user while level 2 contextual outlier explanation may be too abstract.

Based on the limitations we believe a broader study of unsupervised contextual outlier explanations, both during development and for end-users could give valuable information to both the research community and companies that want to use unsupervised contextual outlier detection. We also believe future research should investigate how outlier detection and explanation methods can be further improved via personalization, integration of non-outlier related information, alert correlation, and causal outlier explanation.

These features require closer interaction with domain experts and users, either through mixing in domain rules or retrieving labeled data, e.g., incrementally, with active learning. Such an approach will have its outset in the unsupervised method but will gradually achieve higher precision when labels and domain rules are integrated.

Author Contributions: Conceptualization, J.H.S. and N.D.; methodology, J.H.S. and A.Z.; software, J.H.S. and T.C.; validation, J.H.S., T.C., D.H. and N.D.; formal analysis, J.H.; investigation, J.H.S.; resources, T.C., D.H. and N.D.; data curation, J.H.S.; writing—original draft preparation, J.H.S.; writing—review and editing, J.H.S., A.Z. and P.S.-K.; visualization, J.H.S.; supervision, P.S.-K. and A.Z.; project administration, J.H.S. and N.D.; funding acquisition, J.H.S. and N.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by DigitaliseringsBoost.

Acknowledgments: The authors would like to acknowledge Henrik Vistisen, Erhvervshus Midtjylland, for initiating and facilitating the collaboration between Danmark Music Group Ltd. and Department of Mathematics & Computer Science, University of Southern Denmark.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Moscato, V.; Picariello, A.; Sperli, G. An emotional recommender system for music. *IEEE Intell. Syst.* **2020**, 1. [CrossRef]
2. Amato, F.; Moscato, V.; Picariello, A.; Sperlí, G. Recommendation in Social Media Networks. In Proceedings of the 2017 IEEE Third International Conference on Multimedia Big Data (BigMM), Laguna Hills, CA, USA, 19–21 April 2017; pp. 213–216. [CrossRef]
3. Hochreiter, S.; Schmidhuber, J. Long Short-term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
4. Ribeiro, M.T.; Singh, S.; Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *arXiv* **2016**, arXiv:1602.04938. [CrossRef]
5. Mehrmolaei, S.; Keyvanpourr, M.R. A Brief Survey on Event Prediction Methods in Time Series. In *Artificial Intelligence Perspectives and Applications*; Silhavy, R., Senkerik, R., Oplatkova, Z.K., Prokopova, Z., Silhavy, P., Eds.; Springer: Cham, Switzerland, 2015; pp. 235–246. [CrossRef]
6. Guralnik, V.; Srivastava, J. Event Detection from Time Series Data. In Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 15–18 August 1999; pp. 33–42. [CrossRef]
7. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly Detection: A Survey. *ACM Comput. Surv.* **2009**, *41*. [CrossRef]
8. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly Detection for Discrete Sequences: A Survey. *IEEE TKDE* **2012**, *24*, 823–839. [CrossRef]
9. Schubert, E.; Weiler, M.; Zimek, A. Outlier Detection and Trend Detection: Two Sides of the Same Coin. In Proceedings of the IEEE International Conference on Data Mining Workshop (ICDMW), Atlantic City, NJ, USA, 14–17 November 2015. [CrossRef]
10. Gupta, M.; Gao, J.; Aggarwal, C.C.; Han, J. Outlier Detection for Temporal Data: A Survey. *IEEE Trans. Know. Data Eng.* **2014**, *26*, 2250–2267. [CrossRef]
11. Schubert, E.; Zimek, A.; Kriegel, H.P. Local Outlier Detection Reconsidered: A Generalized View on Locality with Applications to Spatial, Video, and Network Outlier Detection. *Data Min. Knowl. Disc.* **2014**, *28*, 190–237. [CrossRef]
12. Malhotra, P.; Ramakrishnan, A.; Anand, G.; Vig, L.; Agarwal, P.; Shroff, G.M. LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. *arXiv* **2016**, arXiv:1607.00148.
13. Provotar, O.I.; Linder, Y.M.; Veres, M.M. Unsupervised Anomaly Detection in Time Series Using LSTM-Based Autoencoders. In Proceedings of the 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine, 18–20 December 2019; pp. 513–517. [CrossRef]
14. Markou, M.; Singh, S. Novelty Detection: A Review—Part 2: Neural Network Based Approaches. *Signal Process.* **2003**, *83*, 2499–2521. [CrossRef]
15. Miljković, D. Review of novelty detection methods. In Proceedings of the 33rd International Convention MIPRO, Opatija, Croatia, 24–28 May 2010; pp. 593–598.
16. Hyndman, R.; Athanasopoulos, G. *Forecasting: Principles and Practice*, 2nd ed.; OTexts: Melbourne, VIC, Australia, 2018.
17. Siami-Namini, S.; Tavakoli, N.; Namin, A.S. A Comparison of ARIMA and LSTM in Forecasting Time Series. In Proceedings of the 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 1394–1401. [CrossRef]
18. Malhotra, P.; Vig, L.; Shroff, G.M.; Agarwal, P. Long Short Term Memory Networks for Anomaly Detection in Time Series. In Proceedings of the European Symposium on Artificial Neural Networks (ESANN 2015), Bruges, Belgium, 22–24 April 2015.
19. Gunning, D.; Aha, D. DARPA's Explainable Artificial Intelligence Program. *AI Mag.* **2019**, *40*, 44–58. [CrossRef]
20. Lipton, Z.C. The Mythos of Model Interpretability: In Machine Learning, the Concept of Interpretability is Both Important and Slippery. *Queue* **2018**, *16*, 31–57. [CrossRef]
21. Goebel, R.; Chander, A.; Holzinger, K.; Lecue, F.; Akata, Z.; Stumpf, S.; Kieseberg, P.; Holzinger, A. Explainable AI: The New 42? In *Machine Learning and Knowledge Extraction*; Holzinger, A., Kieseberg, P., Tjoa, A.M., Weippl, E., Eds.; Springer: Cham, Switzerland, 2018; pp. 295–303.
22. Knorr, E.M.; Ng, R.T. Finding Intensional Knowledge of Distance-Based Outliers. In Proceedings of the International Conference on Very Large Data Bases, Edinburgh, UK, 7–10 September 1999; pp. 211–222.
23. Zhang, J.; Gao, Q.; Wang, H.H. A Novel Method for Detecting Outlying Subspaces in High-dimensional Databases Using Genetic Algorithm. In Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), Hong Kong, China, 18–22 December 2006; pp. 731–740. [CrossRef]
24. Duan, L.; Tang, G.; Pei, J.; Bailey, J.; Campbell, A.; Tang, C. Mining outlying aspects on numeric data. *Data Min. Knowl. Disc.* **2015**, *29*, 1116–1151. [CrossRef]
25. Kriegel, H.P.; Schubert, M.; Zimek, A. Angle-Based Outlier Detection in High-dimensional Data. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 11–14 August 2008; pp. 444–452. [CrossRef]
26. Kriegel, H.P.; Kröger, P.; Schubert, E.; Zimek, A. Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data. In Proceedings of the 13th Pacific-Asia Knowledge Discovery and Data Mining Conference, Bangkok, Thailand, 27–30 April 2009; pp. 831–838. [CrossRef]

27. Dang, X.H.; Micenková, B.; Assent, I.; Ng, R. Local Outlier Detection with Interpretation. In Proceedings of the ECML-PKDD 2013: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Prague, Czech Republic, 23–27 September 2013; pp. 304–320.
28. Dang, X.H.; Assent, I.; Ng, R.T.; Zimek, A.; Schubert, E. Discriminative Features for Identifying and Interpreting Outliers. In Proceedings of the IEEE International Conference on Data Engineering (ICDE 2014), Chicago, IL, USA, 31 March–4 April 2014; pp. 88–99. [CrossRef]
29. Hawkins, D. *Identification of Outliers*; Chapman and Hall: London, UK, 1980.
30. Zimek, A.; Filzmoser, P. There and back again: Outlier detection between statistical reasoning and data mining algorithms. *Data Min. Know. Discov.* **2018**, e1280. [CrossRef]
31. Rousseeuw, P.J.; Hubert, M. Robust statistics for outlier detection. *Data Min. Know. Discov.* **2011**, *1*, 73–79. [CrossRef]
32. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
33. Tibshirani, R. Regression Shrinkage and Selection Via the Lasso. *J. R. Stat. Soc. Ser. B* **1996**, *58*, 267–288. [CrossRef]