

Detecting Stale Data in Wikipedia Infoboxes

<p>Malte Barth Hasso Plattner Institute University of Potsdam Germany malte.barth@ student.hpi.de</p>	<p>Tibor Bleidt Hasso Plattner Institute University of Potsdam Germany tibor.bleidt@ student.hpi.de</p>	<p>Martin Büßemeyer Hasso Plattner Institute University of Potsdam Germany martin.buessemeyer@ student.hpi.de</p>	<p>Fabian Heseding Hasso Plattner Institute University of Potsdam Germany fabian.heseding@ student.hpi.de</p>	<p>Niklas Köhnecke Hasso Plattner Institute University of Potsdam Germany niklas.koehnecke@ student.hpi.de</p>
<p>Tobias Bleifuß Hasso Plattner Institute University of Potsdam Germany tobias.bleifuss@hpi.de</p>	<p>Leon Bornemann Hasso Plattner Institute University of Potsdam Germany leon.bornemann@ hpi.de</p>	<p>Dmitri V. Kalashnikov USA dmitri.vk@acm.org</p>	<p>Felix Naumann Hasso Plattner Institute University of Potsdam Germany felix.naumann@hpi.de</p>	<p>Divesh Srivastava AT&T Chief Data Office USA divesh@ research.att.com</p>

ABSTRACT

Today’s fast-paced society is increasingly reliant on correct and up-to-date data. Wikipedia is the world’s most popular source of knowledge, and its infoboxes contain concise semi-structured data with important facts about a page’s topic. However, these data are not always up-to-date: we do not expect Wikipedia editors to update items at the moment their true values change. Also, many pages might not be well maintained and users might forget to update the data, e.g., when they are on holiday.

To detect stale data in Wikipedia infoboxes, we combine correlation-based and rule-based approaches trained on different temporal granularities, based on all infobox changes over 15 years of English Wikipedia. We are able to predict 8.19 % of all changes with a precision of 89.69 % over a whole year, thus meeting our target precision of 85 % as suggested by the Wikimedia Foundation. These results can be used to mark potentially stale information on Wikipedia (on average 3,362 fields per week) for readers and to request an update by community contributors.

1 CHANGES ON WIKIPEDIA

Wikipedia is a free online encyclopedia, which is written and maintained by community contributors. Many pages contain infoboxes, which summarize short facts about the topic of the page. For example, the Wikipedia page for *Premier League* contains an infobox that provides insights about the league, such as the property *Current champions*, which has the value *Manchester City*. Infoboxes follow templates, which pre-define the set of properties for the entity type at hand.

Wikipedia is a community-driven effort and most editors contribute in their free time. In their limited time they not only provide new content, but also maintain the existing content of Wikipedia. Due to the fast growth of Wikipedia, the effort for the latter is intensifying, which makes automated support in the editing process highly desirable. Here, we present a system that can support editors by hinting at likely outdated information that needs their attention. This allows it to be updated more quickly and the editors to work more effectively.

Our system, which detects out-of-date properties in Wikipedia infoboxes, can help not only editors, but also readers by making them aware of possibly outdated information. For example, a

Handball-Bundesliga


Season	2018–19	<div style="border: 1px solid black; padding: 5px;"> <p>This value might be out of date: "Matches played" changed two days ago and this value has not been updated yet.</p> </div>
Matches played	224	
Goals scored	2,754 (12.29 per match) 	

Figure 1: A mock-up of how our system could inform about potentially outdated values¹.

marker could be placed on outdated information, as it is already common elsewhere on Wikipedia (e.g., for missing sources). A mock-up for a real example¹ our method identified is shown in Figure 1. In the best case, users then not only check the provided information through external sources, but also update Wikipedia directly, if necessary.

For such a system to be useful, it needs to have a high precision: raising false alerts is both annoying and detracting. We reached out to the Wikimedia Foundation², and in a personal conversation they revealed that such a system should have a precision of at least 85 %. Simultaneously, it needs a noticeable recall (even if it is not very high) to have an impact on Wikipedia infoboxes. Moreover, the system should be able to predict stale data on different time granularities: When a property usually changes daily, e.g., the *num_episodes* of current soap operas, marking it stale on the next day, when we expected a change but it did not occur, is reasonable. In contrast, a daily prediction for a property that changes rarely is more difficult, and we might raise an alert too early. In such cases, predicting stale data on a weekly or monthly granularity makes more sense. Finally, the system must be able to make predictions for all of Wikipedia infoboxes, every day. Since Wikipedia is constantly evolving, e.g., new pages and infoboxes are created, the system needs to be updated regularly. Combined, this results in tight limits on training and prediction time.

A simple baseline, e.g., always predicting the next change based on the mean time-to-change of older changes of that property, does not solve the problem satisfactorily. For instance, it does not work for seasonal data, such as football league seasons. Another natural idea is to use time series forecasting methods to make predictions. However, these methods are not applicable for two reasons: First, most of the data is very sparse, with many

© 2023 Copyright held by the owner/author(s). Published in Proceedings of the 26th International Conference on Extending Database Technology (EDBT), 28th March–31st March, 2023, ISBN 978-3-89318-092-9 on OpenProceedings.org. Distribution of this paper is permitted under the terms of the Creative Commons license CC-by-nc-nd 4.0.

¹<https://en.wikipedia.org/?oldid=889230889>

²<https://wikimediafoundation.org>

properties changing only a few times throughout their lifetime. Second, many of the properties that do change frequently have an irregular change behavior, such as the goals scored by a football player.

To tackle this problem, we present a two-step approach to detect different kinds of change patterns. We detect simultaneously changing properties

- (1) on page level via discovering correlated fields on a page,
- (2) and on infobox template level via association rule mining.

Figure 2 shows examples of both correlations and rules – their content is explained later. Then, we combine the results of the individual approaches by an OR-ensemble to obtain the final result. The field correlation and association rules have another key advantage: they inherently give an explanation for their prediction. We contribute

- (1) a framework for predicting out-of-date data on Wikipedia infoboxes for different time intervals,
- (2) two different change-detection models (field correlation, association rules) for infobox and schema-level data aspects for Wikipedia infoboxes,
- (3) an extensive evaluation of our method on different time granularities, and
- (4) a combination of the models, achieving the performance level desired by Wikimedia Foundation.

The structure of the paper is as follows. First, we discuss related work in Section 2. Next, we describe the problem setting, our two change prediction models, namely field correlations and association rules, as well as ensemble methods in Section 3. Section 4 describes the dataset that we use for the evaluation of our approach in Section 5. Finally, we draw conclusions and provide ideas for future work in Section 6.

2 RELATED WORK

All revisions of Wikipedia pages are freely available [14] and have been extensively researched, as Mesgari et al. have surveyed [7]. According to them, the main research areas focus on the size of Wikipedia or on its quality. In this paper, we consider the latter.

Data Quality. Given the size and the manual updates of Wikipedia, data quality inevitably becomes an issue. Fürber and Hepp [4] suggest a framework for information quality assessment of Semantic Web data that can be used to automatically identify deficient data. To achieve this, they provide a set of generic SPARQL queries to detect missing values or functional dependency violations.

Tran and Cao [11] propose a novel method to automatically detect outdated attribute values in Wikipedia infoboxes by using pattern-based fact extraction with facts extracted from the web, achieving an accuracy of 77%. They improved their own approach [12] by combining it with an entity-search-based approach for an accuracy score of 82%. However, both of their accuracy numbers were shown only for one specific attribute within a specific infobox, namely the number of employees in a company, which makes an assessment of their general accuracy difficult. In our approach, we aim to detect stale data in *all* attributes.

Data Cleaning. There have also been novel ideas to not only detect stale data but also clean it as well. Milani et al. [8] propose the probabilistic CurrentClean system, which can learn the update patterns of a database and then infer the actual updates the database ought to make. Wang and Wang [13] evaluate and compare many state-of-the-art techniques for data cleaning in

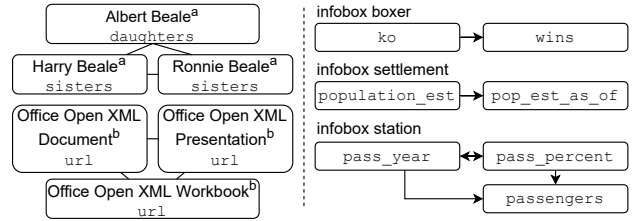


Figure 2: Examples of field correlations (left) and template level association rules (right). ^a<https://en.wikipedia.org/?oldid=1121049020> ^b<https://en.wikipedia.org/?oldid=1111262597>

a time-series context. They conclude that most techniques are developed for relational databases and not time-series data like in our scenario. Given these results and the scope of our work, we focus on the detection of stale data, and leave its cleaning to future work.

Association rule mining. One possibility to find related properties is frequent itemset or frequent pattern mining. With association rule mining, it is possible to discover potential causal relationships between items of a frequent itemset. At the heart of these fields of study lie algorithms that efficiently discover frequent patterns, such as the Apriori algorithm [1], which efficiently traverses and prunes a lattice of candidate itemsets before generating association rules from them. This is also the algorithm that we employ. Another related field is *frequent episode mining* [6], which, in contrast to association rule mining, does not mine rules on transactions of events, but instead on a single event sequence. This makes it unsuitable for the scope of Wikipedia, as modelling changes to Wikipedia as a single sequence would lead to a prohibitively large number of event types. CR-Miner [5] also searches for rules between change events, but aims at finding *rare and exceptional* relationships. In contrast, we are interested in frequently occurring associations because they have the greatest potential to reveal stale data.

3 CHANGE PREDICTORS

As explained in Section 1, there is a high variance in the change patterns that we can find in Wikipedia. Thus we tackle the problem of detecting stale data using two algorithms that each focus on a different aspect of the data. First, we define the problem that we address in this paper in Section 3.1. Then we propose a distance-based correlation measure to detect correlated field change histories in Section 3.2, and an association rule mining approach to detect patterns within infobox templates in Section 3.3. Finally, we combine these approaches into an ensemble to cover various aspects of the data effectively, as described in Section 3.4.

3.1 Preliminaries and Problem

As a model to represent the changes on Wikipedia, we use the change-cube [2]. It describes four dimensions of each *change* in the data: *time* (when did the change happen), *entity* (the unique id of the infobox to which the change belongs), *property* (specific attribute of an entity) and *value* (the newly assigned value). We call combinations of entity and property *fields*. Additionally, our data contains shared sets of properties for groups of infoboxes, which are referred to as *templates*. Each infobox (entity) belongs to exactly one template. As we want to predict when the next change will happen and not what the new value shall be, we can disregard the value dimension for our approaches.

We solve the following problem: Given the current time t , a time window size w and a candidate field f , that did not change in $[t - w, t]$, predict whether f *should* have changed in $[t - w, t]$.

In our experiments, we choose $w \in \{1d, 7d, 30d, 365d\}$ to reflect the usual calendar time periods. As this is a binary classification task, we use precision and recall to measure the model performance and optimize recall given a target precision of 0.85.

3.2 Field Correlations

Changes in different fields can occur in close temporal proximity. Semantically linked fields, i.e., those that represent the same underlying concept, are one reason. For example, if a soccer club experiences an update to its uniform home colors, it should also receive an update to the away colors, as these are usually changed in unison once per season.

Our field correlation predictor is based on rules that represent semantically related fields. The rules are used to identify stale data and represent implications of changes over time: Given fields X and Y , the field correlation rule $X \sim Y$ states that a change in X should also imply a change in Y during the same time period and vice-versa. We tried different time periods (to, e.g., allow delayed updates), but same-day worked best on our dataset. Field correlation rules contain one field on both the left- and right-hand side (but larger clusters can be modeled through multiple rules).

We represent the change history of a field P as a k -dimensional vector v , which represents the number of changes per day of the field for each of the k days that is present in the training data. We create the set \mathcal{V} of vectors, which contains a vector for each field. We then define an error metric $M : \mathcal{V} \times \mathcal{V} \mapsto [0, 1]$, which measures how uncorrelated two fields are: M is the Manhattan-distance normalized by the vector length k . Here, 0 represents a perfect correlation (they always change together) and 1 indicates no overlapping changes. To find correlated fields in a set of fields, we use this metric to calculate the pairwise distance between vectors of this set. Two fields are part of a correlation rule if the metric M for their two vectors is lower than the error threshold θ (see Section 5.2 how we tuned this threshold).

With a quadratic runtime $O(k|\mathcal{V}|^2)$, it is computationally infeasible to calculate *all* distances between members of \mathcal{V} . Furthermore, in a large set of fields, many spurious correlations are likely to occur. Thus, we restrict the correlation search to fields of the same page to avoid this problem and reduce the complexity to $O(k \sum |\mathcal{V}_i|^2)$ where \mathcal{V}_i is the set of vectors of all fields on page i . We originally performed experiments using changes across multiple pages, for instance including all pages that are linked from the page under consideration. Analyzing correlation proved infeasible, and we aborted the analysis after one week. In another variant, we included only symmetrically linked pages. Here, the recall increased only in the third decimal digit. Figure 2 shows examples of field correlations that we have found in the data: the daughters of a series character should change in the same way as the sisters of his sons change and furthermore, the urls of various Open Office formats change synchronously.

To make a prediction if a field should have changed, we check whether a field has correlated fields, meaning it is part of a field correlation rule. Additionally, we require a time window for which we want to make a prediction. For all rules that contain the field of interest, we check all other contained fields for changes inside the given time window. Should there be any change, we make a prediction that the field should have changed within that

time window. If no change in the target field can be observed, the field can be marked as stale.

3.3 Association Rules

While the field correlation approach excels in finding rules specific to pairs of fields, it falls short with entities and properties that change rarely or were only recently created, because the information an individual field can provide for such cases is limited. Thus, we introduce a second approach that aims to capture relationships that hold for groups of similar entities, and apply them to predict stale data for all members of these groups, regardless of whether a particular instance of a group was present in the training data. We capture these relationships between properties of groups of entities – here, all infoboxes that belong to the same template – by finding association rules between them. The example dependency of a soccer club’s season, between `goals_scored` and `matches_played` illustrates the differences to the previous approach: (1) the relationship between `goals_scored` and `matches_played` is not symmetric – if a match is played, a goal is not guaranteed, and (2) it should hold for all soccer clubs’ seasons, not only for a particular one.

Before discovering association rules that capture relationships on the input data (tuples of the change-cube), we must partition it into transactions and assign the changes to an item (event type) so that we can mine rules on the transactions containing these items. We assemble transactions by gathering all changes that were made across Wikipedia during some predefined time period. If this period is very short, such as a few nanoseconds, the resulting association rules describe relationships that hold only for this very short period, resulting in few association rules. On the other hand, if the period is very long, such as a few years, the resulting association rules are not useful in our use case: predicting a change anytime in this period would not help in finding stale information in a timely manner. To reflect the expected editing workflow of volunteer Wikipedia contributors, which we assume to be weekly [15], we set this period to a granularity of 7 days. This is done for each infobox, resulting in a transaction for each week and infobox combination.

To assign changes to an event type, we have, in principle, five dimensions available per change: the *time*, the *entity*, the *property*, the *value* and the *template*. Including the timestamp in the definition of an event type would result in several changes to the same infobox field being represented as different event types, thus leading to association rules that hold only for specific timestamps. Because we aim to find association rules that hold at all timestamps, we do not include the time in the event types. Second, considering the actual values would limit the resulting rules to the domain of historical values. Since we do not expect to see all (or even a fraction) of possible future values in our historical dataset, we also exclude the value dimension. Finally, including the entity ID would lead to rules that are specific to certain entities, which is not what we desire here, because this functionality is already covered by the field correlations as described in Section 3.2. Thus, we only use the property and the template of the changed infobox to identify the event type of a change. This allows us to discover general rules that capture relationships that hold for all infoboxes of a template.

With the transformed data as a transaction list, we use the Apriori algorithm [1] (we describe its configuration in Section 5.2) to find unary association rules. The complexity of transaction generation is $O(n)$, where n is the number of changes, and that of

rule mining is $O(m^2 \cdot |T|)$, where m is the number of properties and $|T|$ is the number of transactions. After mining association rules, we use a subset of the training data that we were holding out to validate the precision of the rules. Note that the test data was removed beforehand. Using our target of 85% precision with a 5% buffer to account for performance decrease between the training and test set, we discard rules that do not meet 90% precision on the validation set. We make a prediction for an infobox if one of its properties changes and there is an association rule for the template of the infobox where the property that changed is at the left-hand side of the association rule.

Figure 2 shows examples of association rules that we found with this approach, such as the dependency between knockouts and wins for all infoboxes of boxers, or the population estimate and its date for settlements. We note in these particular examples that the relationship between wins and knockouts should indeed be asymmetric (a change in property `ko` should entail a change in property `wins`, but not vice versa), as correctly captured by the association rule approach, thus complementing the field correlation approach that is restricted to symmetric relationships.

3.4 Ensemble

The associations and field correlation rules both describe ways to label fields as outdated, but focus on different aspects of the data. As both predictors are optimized for roughly the same target precision, we can combine the predictions by disjunction to boost our recall while expecting to keep a high precision. We call this combination the OR-ensemble. It is also possible to boost the precision at the cost of recall by conjunction, which we call AND-ensemble. We evaluate how different the predictions of the field correlation and association rule approaches are in Section 5.3.4.

4 DATASET

The entire history of infoboxes of all English Wikipedia pages from January 4, 2003 until September 2, 2019 has been exported by Bleifuß et al. [3]. It consists of the state of Wikipedia infoboxes over multiple revisions as they were edited by the contributors, with a total of 283 million changes. The Wikipedia infoboxes contain the information in a key-value pair structure. For instance, the infobox of the page *Unites States* contains the property `largest_city` with the value `New York City`. Every infobox has exactly one infobox template, which describes how properties are displayed. For example, the infobox of London has the template `Infobox settlement`.

Due to the nature of the data, the dataset contains a lot of noise caused by edit wars, typo fixes and vandalism [9, 10]. Before we train and evaluate our models, we filter the data to remove noise and data that is uninteresting for our use-case, by applying a set of filters to each change history separately. Some filters adjust the change history, while others discard whole change histories. After applying these filters, 9.2% (25 million) of the original changes remain. All our change predictors use subsets of this filtered data for training and testing. We briefly describe the filters in the following.

A small number of all changes (0.008%) are directly reverted by Wikipedia bots. This usually includes accidental edits or vandalism. As these changes do not have any value for us – they do not indicate that a property value was updated – we filter them out. To limit the influence of vandalism further, we reduce the resolution of the time dimension by grouping all changes on a day for every change history to one representative change,

eliminating 19.185% of the remaining changes. A representative change is defined as the *mode* of all values that the property had over the day. We keep the most recent value in case of ties.

Furthermore, a large part of the data are creations (50.6%) and deletions (20.3%), which we do not aim to predict and are not helpful for our models. A property is created when a new infobox is created or a property is added. On the other hand, a property is deleted if the infobox it belongs to is deleted or a single property of an infobox is removed. Ignoring creations and deletions removes another 61.373% of the remaining changes.

Finally, there are also many properties that should never change. For instance, infoboxes of people usually contain the properties `birth_date` and `birth_name`, which should not be subject to any changes. Thus, the underlying field can be seen as static, and is not of interest for the problem of change prediction which is why we filter changes of fields with fewer than five changes (10.241%) to focus on the more frequently changing part of Wikipedia. While this filter creates a bias towards dynamic data, we require some historical data for training of the field correlation predictor. To keep the individual results comparable, we applied this filter for all predictors. However, experiments showed, that the association rules, which can work on data without any changes, achieve similar precision without this filter.

5 EVALUATION

In the following, we describe our evaluation of our proposed change predictors. We describe two baselines and simulate predictions over a 365-day time span on different granularities. We evaluate these predictions on mean precision and mean recall, and provide more detailed analyses, such as precision over time and between groups of fields with varying degrees of change frequencies. Our code is available on GitHub³.

5.1 Experimental Setup

To evaluate the different change predictors, we split all available data into training, validation and test sets. All approaches are trained on the training set, optimized on the validation set, then trained on both training and validation set and finally evaluated on the test set. The ensembles require no optimization and are thus applied to predictors trained on the training and validation sets.

The train, validation, and test splits are made along the time axis rather than the field axis: all datasets contain all fields that have at least five changes within their timeframe. The test set starts September 1, 2018 and spans 365 days. The validation set is marked as the 365 days *before* the start of the test set. The training set contains all filtered data up to the validation set. This results in a training set of 4,835 days beginning June 5, 2004, and a validation and test set of 365 days each.

Predictions are made on field level at various degrees of granularity. The dataset to be predicted is split into 1-, 7-, 30- and 365-day tumbling windows. Windows that would exceed the validation and test set limit are disregarded (which is the case for the last 7- and 30-day windows). For each window and for each field, a prediction has to be made, which states whether a field should change within the given time window. Predicting on all granularities results in 430 predictions ($365 \times 1d + 52 \times 7d + 12 \times 30d + 1 \times 365d$) per field for both the validation and test set. The overall training and prediction time is about 6 hours on a Dell PowerEdge R810 with four Intel Xeon E7-8837 and 256GB of RAM.

³https://github.com/HPI-Information-Systems/wikipedia_cleanup

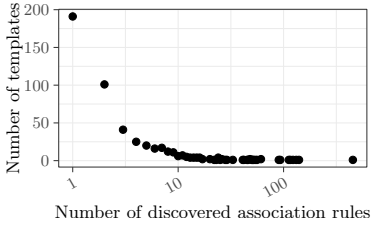


Figure 3: Number of association rules (logarithmic x-scale) discovered for infobox templates.

	1 day			7 days			30 days			365 days		
	P [%]	R [%]	#	P [%]	R [%]	#	P [%]	R [%]	#	P [%]	R [%]	#
Mean baseline	4.69	1.86	887,192	13.22	6.16	891,206	21.37	12.12	838,415	51.47	34.33	521,777
Threshold baseline	0.00	0.00	0	80.77	0.06	1,456	60.47	0.45	11,016	53.59	57.24	835,791
Field correlations	87.66	5.19	132,537	88.74	4.99	107,715	88.20	3.96	66,442	90.55	3.19	27,599
Association rules	91.73	5.63	137,436	93.30	5.35	109,890	93.43	4.60	72,804	95.52	3.86	31,594
AND-ensemble	96.08	2.31	53,803	96.58	2.16	42,738	96.68	1.77	27,129	98.06	1.46	11,666
OR-ensemble	88.16	8.51	216,173	89.69	8.19	174,829	89.54	6.79	112,084	92.02	5.59	47,513

Table 1: Precision, recall, and absolute number of predictions (true and false positives) of all predictors on the test set.

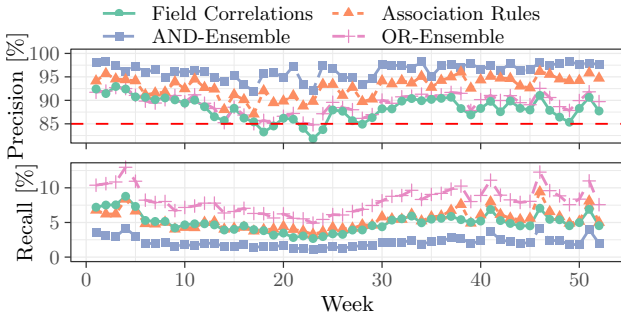


Figure 4: Precision and recall over time of our predictors on 7-day windows on the test set.

For each prediction of a field f in a window w , the change predictors receive the start date of w , all changes to f up to the start date of w , and for other fields $f' \neq f$, the predictors receive all changes before and within w . This simulates a realistic scenario in which one edit for field f was forgotten by the editors, but related fields were updated correctly.

Predictions are evaluated separately for each of the four window sizes. We analyze the resulting binary classification problem in terms of precision and recall. Specifically, we focus on maximizing recall while keeping precision above a previously set threshold of 85%. Such a threshold would be necessary to deploy the mentioned predictors in a recommendation setting, as described in Section 1. Additionally, we report the number of predicted changes to give a better intuition on the real-world impact of our solution.

5.2 Baselines and Predictor Configuration

We compare our predictors to two baselines. The first baseline is a regressor that creates predictions by stating a next change will happen in the next n days, starting from the last known prediction. We determine n as the mean number of days between changes for each field separately. We call this the *mean baseline*. This regression is then converted to a classification by checking whether the day of the next estimated change falls within a window of interest.

The second baseline makes use of basic statistical properties and is called *threshold baseline*. When training, we check if a field had a change in more than a specific threshold of windows of a window size during the validation set period. We use the previously mentioned threshold of 85%. If this is the case, the baseline predicts a change for every window of that size during testing. For example, if a field changed in at least 45 (85% of 52)

7-day windows, we predict a change for all 7-day windows in the test set.

For our field correlation predictor, we perform a grid search over suitable error thresholds θ ranging from 0.01 to 0.15. We choose 0.1, as it provides the highest recall (5.19%) while maintaining a precision above our threshold (87.65%). The 2.65% margin between this precision and our threshold is desirable, as we expect precision on the test set to drop slightly because the validation set may differ from the test set in terms of data distribution.

For the association rule predictor, we ran a thorough grid search on parameters of the Apriori algorithm (min-confidence and min-support) and the size of the validation set used to discard rules with a low precision. The results of the predictor are evaluated on the validation set. We arrive at a min-support of 0.25%, a min-confidence of 60% and a validation set size of 10%. Figure 3 shows the number of rules discovered for the infobox templates: The association rule predictor discovers a total of 3,852 association rules, which cover 248,865 pages in our dataset. The distribution among templates is skewed: for 191 of the 8,276 templates in the dataset, exactly one association rule is discovered each, but there is also one template (infobox legislative election) with more than 150 association rules. The number of rules is limited by the number of property pairs, so it is quite surprising that templates with so many rules exist.

5.3 Results

The results of all predictors on the test set are shown in Table 1. The total number of windows containing changes is 2,239,604 for the 1-day, 1,914,466 for the 7-day, 1,478,266 for the 30-day, and 782,304 for the 365-day window. The difficulty of predicting changes for the windows decreases as the window size increases. A perfect prediction on 1-day windows can be used to create perfect predictions on all other window sizes. Thus, we expect that average results on 1- and 7-day windows are worse than on 30- and 365-day windows.

5.3.1 Baselines. Both baselines make predictions solely on the data of the field to be predicted, as opposed to the field correlation and association rule predictors, which require additional data. The mean baseline does not achieve a precision over 55% across all window sizes. Both precision and recall linearly increase as the size of the windows increases. This confirms that smaller window sizes are harder to predict for this baseline. The threshold baseline makes no predictions for the daily prediction because no field had 311 (85% of 365) or more changes in the previous year. Both baselines fail to achieve results that exceed

our precision threshold in any of the different window evaluations. They do, however, have the highest recall on the 365-day windows by a large margin.

5.3.2 Our predictors. The field correlation with the configuration described in Section 5.2 performs just marginally better in terms of precision on the test set compared to the validation set on the 1-day window. The precision is up to 0.2% higher, whereas the recall is equal. Similar results can be found for all four tested window sizes, indicating that the additional year of training data does not have a large effect on the field correlations. We also conclude that the data distribution of the validation and the test sets are similar. Precision on all window sizes is above our defined target precision.

Using the best configuration found with the grid search, the association rule predictor achieves a precision of 91.73% on the test set with a recall of 5.63% at a daily prediction interval with a total number of 137,436 predictions within the test set spanning 365 days. The precision increases as the prediction interval increases while the recall and total number of predictions decrease. At a yearly prediction interval, we observe a precision of 95.52%, a recall of 3.86% and a total number of 31,594 predictions. These results are marginally better in terms of precision than on the validation set used to perform the grid search, where we observe a precision of 91.08% and a recall of 4.92% at a daily prediction interval. Overall, recall is low as expected, as we measure it against *all* changes to all Wikipedia infoboxes, most of which are likely to occur in a non-systematic fashion or on whims of the editors. The ability to reliably predict even a small percentage of changes translates to large quantities of correct change-predictions.

5.3.3 Behavior over time. The precision over time can be seen in the upper plot of Figure 4. It contains a slight downwards trend as time increases, but remains above the 85% precision threshold at the end of the year. However, it dips below this threshold in the middle of the test set. Since it increases afterwards, and a similar dip can be seen in the association rule predictor, we assume it is not due to problems with an outdated predictor, but rather a particularly irregular period of data. This dip is also present in the recall (lower plot in Figure 4). As properties are renamed and data is deleted and created over time, the percentage of data for which rules exist drops influencing recall. With decreasing precision and slightly decreasing recall, we recommend retraining at least once per year to maintain both high precision and recall. Overall, both predictors show a relatively stable and similar behavior over time.

5.3.4 Ensembles. Lastly, we evaluate the ensembles of the field correlation and association rule predictors (see Table 1). As expected, the AND-ensemble shows higher precision and lower recall than the two predictors, while the OR-ensemble has higher recall and a precision between both. The OR-ensemble is the predictor with the highest recall that also satisfies our prediction threshold, and thus is our best performing predictor. Both predictors create overlap in 37 - 42% of their predictions, meaning 58 - 63% of their predictions are unique and contribute to the recall of the OR-ensemble. This number is quite high given the relatively low recall of both predictors, and shows that the template-based association rules and the field-based field correlation rules partly contain redundant information. However, as both contribute significantly to the success of the OR-ensemble, we believe both approaches have their use.

5.4 Comparison to Ground Truth

In our evaluation, predictors can be penalized for making correct change predictions that should have happened according to real-world events, but in fact were not represented as a change in Wikipedia. Such cases particularly show the potential benefit of our system. Thus, we investigated results on a few selected infobox change histories. The following example shows one of these cases:

For the German Handball-Bundesliga season 2018–2019⁴, we find that we correctly predict three changes to total goals that are not in the change history of the infobox. It is found by an association rule for the template infobox football league season, which is also used for the Handball-Bundesliga seasons. The symmetrical rule states that a change in matches leads to a change in total goals and vice versa. For three different days with played games, matches is correctly updated, but total goals is not.

We also found an interesting behavior for the values of total goals per se. Editors often just incremented the current value of total goals without verifying it. In one instance, the total score of 9,880 changed to 1,073 instead of 10,073 – a typo. This wrong value was then incremented multiple times until the last day of the season, where it was changed from 6,197 to the correct 16,227.

6 CONCLUSIONS

As information on Wikipedia grows at a fast pace, the need to keep information up-to-date arises. Especially in infoboxes, which summarize an entity to consume at a quick glance, correct and up-to-date data is crucial. Using infobox change histories, we examined the staleness of data and propose a solution to inform about stale infobox data. With a successfully reached target precision of 85%, we achieve a recall of 8.19%, which allows us to classify on average 3,362 fields per week as stale. In terms of absolute numbers, this is a significant recall that can make a difference for editors and readers.

We recognize limitations to our approach that shall be addressed in future work. As we rely on simultaneous (same day) changes, we cannot capture cases where the updates of related properties are also delayed or there are no related properties with similar change patterns. A way to tackle this problem is adding predictors to the ensemble that focus on other aspects of the data: they could capture seasonality or consider further signals, such as changes to the text of Wikipedia articles or additions and deletions of values. Grouping annual events like soccer seasons, the Oscars, or the Olympics, where each year has its dedicated Wikipedia page, could help identify more general patterns across less frequently changing pages. Finally, the approaches could be generalized to other datasets containing structured data [3] or even be extended to textual data.

REFERENCES

- [1] Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast algorithms for mining association rules. In *Proceedings of the International Conference on Very Large Databases (VLDB)*. Morgan Kaufmann, 487–499.
- [2] Tobias Bleifuß, Leon Bornemann, Theodore Johnson, Dmitri V. Kalashnikov, Felix Naumann, and Divesh Srivastava. 2018. Exploring Change: A New Dimension of Data Analytics. *PVLDB* 12, 2 (2018), 85–98.
- [3] Tobias Bleifuß, Leon Bornemann, Dmitri V. Kalashnikov, Felix Naumann, and Divesh Srivastava. 2021. Structured Object Matching across Web Page Revisions. In *Proceedings of the International Conference on Data Engineering (ICDE)*. IEEE, 1284–1295.
- [4] Christian Fürber and Martin Hepp. 2011. Swiqa—a semantic web information quality assessment framework. In *ECIS 2011 Proceedings*. 76. 76.

⁴https://en.wikipedia.org/wiki/2018-19_Handball-Bundesliga

- [5] Daniel Lindner, Franziska Schumann, Nicolas Alder, Tobias Bleifuß, Leon Bornemann, and Felix Naumann. 2022. Mining Change Rules. In *Proceedings of the International Conference on Extending Database Technology (EDBT)*. 91–103.
- [6] Heikki Mannila, Hannu Toivonen, and A Inkeri Verkamo. 1997. Discovery of frequent episodes in event sequences. *Data mining and knowledge discovery* 1, 3 (1997), 259–289.
- [7] Mostafa Mesgari, Chitu Okoli, Mohamad Mehdi, Finn Årup Nielsen, and Arto Lanamäki. 2015. “The sum of all human knowledge”: A systematic review of scholarly research on the content of Wikipedia. *Journal of the Association for Information Science and Technology* 66 (2015), 219–245.
- [8] Mostafa Milani, Zheng Zheng, and Fei Chiang. 2019. CurrentClean: spatio-temporal cleaning of stale data. In *Proceedings of the International Conference on Data Engineering (ICDE)*. IEEE, 172–183.
- [9] Martin Potthast and Teresa Holfeld. 2010. Overview of the 1st International Competition on Wikipedia Vandalism Detection. In *CLEF*. CEUR-WS.org.
- [10] Róbert Sumi, Taha Yasseri, András Rung, András Kornai, and János Kertész. 2011. Edit Wars in Wikipedia. In *IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing*. 724–727.
- [11] Thong Tran and Tru H Cao. 2013. Automatic Detection of Outdated Information in Wikipedia Infoboxes. *Res. Comput. Sci.* 70 (2013), 211–222.
- [12] Thong Tran and Tru H Cao. 2013. A hybrid method for detecting outdated information in Wikipedia infoboxes. In *The International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for Future (RIVF)*. IEEE, 97–102.
- [13] Xi Wang and Chen Wang. 2019. Time series data cleaning: A survey. *Ieee Access* 8 (2019), 1866–1881.
- [14] Wikimedia. 2023. Wikimedia Downloads. <https://dumps.wikimedia.org/> [Online; accessed 4-January-2023].
- [15] Taha Yasseri, Robert Sumi, and János Kertész. 2012. Circadian patterns of wikipedia editorial activity: A demographic analysis. *PLoS one* 7, 1 (2012), e30091.