Learning to Predict the Departure Dynamics of Wikidata Editors

Guangyuan Piao[1] and Weipeng Huang[2]

[1] Department of Computer Science, Maynooth University, Maynooth, Ireland
guangyuan.piao@mu.ie
[2] Insight Centre for Data Analytics, University College Dublin, Dublin, Ireland
weipeng.huang@insight-centre.org

Abstract. Wikidata as one of the largest open collaborative knowledge bases has drawn much attention from researchers and practitioners since its launch in 2012. As it is collaboratively developed and maintained by a community of a great number of volunteer editors, understanding and predicting the departure dynamics of those editors are crucial but have not been studied extensively in previous works. In this paper, we investigate the synergistic effect of two different types of features: statistical and pattern-based ones with DeepFM as our classification model which has not been explored in a similar context and problem for predicting whether a Wikidata editor will *stay* or *leave* the platform. Our experimental results show that using the two sets of features with DeepFM provides the best performance regarding *AUROC* (0.9561) and *F1* score (0.8843), and achieves substantial improvement compared to using either of the sets of features and over a wide range of baselines.

Keywords: Wikidata editors · Crowdsourcing dynamics · Classification.

1 Introduction

Wikidata [28] is a community-driven knowledge base with its initial primary goal to be a central knowledge base to serve all Wikimedia projects. Since its launch in late 2012, it has become one of the most active projects of the Wikimedia Foundation in terms of contributors [13]. As one of the largest open, free, multilingual knowledge bases, Wikidata has contributed significantly to the Linked Open Data Cloud[3][4] and our research community along with DBpedia [11]. Wikidata currently contains over 90M items and over 13B edits have been made since the project launch[5], and has been widely used in various domains such as natural language processing [10], recommender systems [23], and life sciences [29].

As a community effort, editors on open collaborative knowledge bases such as Wikidata and Wikipedia add and edit information collaboratively and play a crucial role in the growth of those platforms. Therefore, in the context of

[3] https://lod-cloud.net/
[4] https://bit.ly/2O1KZAV
[5] https://www.wikidata.org/wiki/Wikidata:Statistics/en

Wikipedia, there have been many studies on predicting the editing dynamics of users[6] such as how many contributions an editor will make or whether the editor will stay on or leave the platform [31,2]. Similar to Wikipedia, editors on Wikidata platform are critical to its success and understanding editing dynamics such as whether an editor will leave the platform is important but little attention has been given in the context of Wikidata [24], which is our focus of this study.

We seek answers to questions such as (1) what types of features are useful for predicting the departure dynamics of Wikidata editors, (2) what types of machine learning (ML) approaches perform well for the prediction, and (3) how well those approaches applied to Wikipedia can perform in the context of Wikidata. To this end, we investigate different types of features and investigate a wide range of ML approaches including best-performing ones adapted from previous studies in the context of Wikipedia, and investigate a new approach adopted from the recommender system domain which has not been explored in previous studies for predicting whether an editor will stay on or leave Wikidata. In summary, our contributions are: (1) We investigate two sets of features – statistical and pattern-based ones – for predicting inactive editors on Wikidata (Section 3). We show the synergistic effect on using both sets of features in Section 6, which has not been used together in a similar context; (2) We adopt DeepFM model [9] as our classification approach, which is exploited for the prediction task in the context of Wikidata for the first time to our best knowledge, and show its effectiveness by comparing a wide range of classification models in Section 6 including those applied to Wikipedia; (3) Our source code and the processed Wikidata dataset can be found here[7]. The dataset includes more than 5B edits by 371,068 users (Section 4), which can be a good resource for studying different problems such as recommending Wikidata items.

2 Related Work

In the context of Wikipedia, many studies have been conducted regarding the departure dynamics of Wikipedia editors in the literature. For example, Gandica et al. [7] defined a function of edit probability and showed that the editing behavior of Wikipedia editors is far from random and the number of previous edits is a good indicator of their future edits. Zhang et al. [31] proposed a ML approach with a set of statistical features extracted from different periods of each user's edit history for predicting the future edit volume of Wikipedia editors where they showed that GBT (Gradient Boosted Trees) and kNN (k-Nearest Neighbors) provide the best performance. Instead of *statistical* features, Arelli et al. [1,2] proposed leveraging *pattern-based* features with respect to consecutive edited pages, and constructed Boolean features regarding a set of frequent patterns for predicting whether a Wikipedia user might leave or stay on the platform. In addition to predicting activeness or edit volume of users, other aspects regarding edit behaviors such as quality, edit sessions on the platform, and

[6] We use the words *editors* and *users* interchangeably in this rest of the paper.
[7] `https://bit.ly/3yyJhZj`

the difference between Wikipedians and non-Wikipedian editors have been studied as well [4,8,16]. In contrast to the popularity of previous studies regarding different aspects on Wikipedia, less studies have been explored in the context of Wikidata, which is our focus in this study for classifying potential inactive Wikidata editors.

Recently, Mora-Cantallops et al. [14] conducted a literature review of research on Wikidata and revealed the main research topics on Wikidata such as users and their editing practices, knowledge organization, external references, and the language of editors [19,21,10]. We focus on the first topic (users and their editing practices), which is most relevant to our work. Piscopo et al. [20] studied different types of editors such as bots, human editors, registered, anonymous editors to understand how those editors influence the quality of items on Wikidata. In [15,3] the authors performed a cluster analysis of the editing activities of Wikidata editors to compare them with typical roles found in peer-production and collaborative ontology engineering projects, and studied the dynamic participation patterns across those characterized roles of Wikidata editors. More recently, Sarasua et al. [24] conducted a large-scale longitudinal data analysis for Wikidata edit history over around four years until 2016 to study the evolution of different types of Wikidata editors. Their study revealed many interesting findings such as the number of new editors joining the Wikidata has been increasing over time, and the majority of contribution has been made by a few editors with a skewed distribution of contributions. We observe that similar trends have continued in our Wikidata dataset with edit history until 2020 (Section 4). The authors [24] also investigated the edit volume and the lifespan (i.e., short or long) of editors during their active time on Wikidata where their focus is on *gone* editors.

Compared with those works, we focus on the problem of predicting the activeness of registered editors in the future on Wikidata in this work, investigate the synergistic effect of considering both statistical and pattern-based features, and exploit DeepFM model for the first time for the problem.

3 Proposed Approach

This section provides a formal definition of the Wikidata user classification task, followed by the description of the proposed approach.

Problem formulation. Our goal is to learn a binary classifier $f(\mathbf{x}_u) \rightarrow y_u$ where $\mathbf{x}_u$ denotes a set of features based on the edit history of a user u, and y_u is the class label indicating activeness of u with 1 for inactive and 0 for active.

Overview of our approach. The approach for the classification task of Wikidata editors consists of (1) statistical and/or pattern-based features, and (2) a DeepFM classification model where those features are used as an input.

3.1 Statistical Features

Here we discuss the set of statistical features used in our model. These features utilize the edit history of Wikidata editors. All of these features try to capture users' editing behavior on Wikidata from different perspectives.

1. *Total # of edits* ($N_{total \cdot edit \cdot ent}$) indicates the total number of edits have been made by an editor on Wikidata.
2. *Distinct # of edited entities* ($N_{dist \cdot edit}$). The *total # of edits* does not distinguish edited entities. This feature aims to capture the number of distinct entities have been edited by a user.
3. *Diversity of edit actions* ($Div_{edit \cdot act}$). To capture the diversity of different types of edit actions (see Section 4), we use the Shannon-Entropy [25] of different edit actions in the same manner as in [24] as: $H(T) = -\sum_{i=1}^{n} P(t_i) \cdot \log P(t_i)$ where T indicates different types of edit actions, and $|T| = n$.
4. *Diversity of entities* (Div_{ent}). We measure the diversity of edited entities of a user using the Shannon-Entropy. The intuition is that the diversity of edited entities of a user could also be different across active and inactive editors.
5. *The # of days between first and last edits* (T_{edit}) refers to the time difference between a user's first and last edits during a certain period.
6. *Diversity of day of week* (Div_{day}). This measures the diversity of day of the week based on the edit history of each user using the Shannon-Entropy.
7. *The # of days between first edit and prediction time* ($L_{first \cdot pred}$). This indicates the time difference between a user's first edit on Wikidata and the prediction time to predict whether a user will become inactive or not.
8. *The # of days between the last edit and prediction time* ($L_{last \cdot pred}$) indicates the time difference between a user's last edit and the predicted time.

For the first six features, we extract those in the last p months from the prediction time using 10 different time periods $p \in P = \{\frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 12, 36, 108\}$ to capture the editing behavior over different time periods, which has been inspired by [31] in the context of Wikipedia for predicting the edit volume of users. For instance, we extract the set of features based on the edit history from the last 12 months from the prediction time when $p = 12$. Here, 108 months in our Wikidata dataset (see Section 4) cover the whole edit history of each editor. As a result, there are $6 \times 10 + 2 = 62$ statistical features in total. As one might expect, some of those features can be skewed and we apply a logarithmic scale, $x_{after} = \log(x_{before} + 1)$, to them before feeding those into any ML approach.

3.2 Pattern-based Features

In contrast to *statistical* features such as the above-mentioned ones, *pattern-based* features have been explored in the context of Wikipedia for predicting the departure dynamics of editors [1]. To investigate which type of features performs better or the synergistic effect of combining those two sets of features for our classification task on Wikidata, we investigate those statistical and pattern-based features – separately and together – in our experiments in Section 5. In the following paragraphs, we describe the set of pattern-based features used.

To start with, the edit history of each user is considered as a chronological sequence of each consecutive pair (i_1, i_2) of edited entities. For each pair (i_1, i_2), the following information is extracted for describing each pair:

- **r/n**: Whether i_2 is an entity that has already been edited by the user before, i.e., **r** if i_2 is a re-edit, and **n** otherwise.
- **m/n**: Whether i_2 is a *normal entity* (**n**) or others such as *properties* (**m**). On Wikidata, each entity is identified by a unique entity ID, which is a number prefixed by a letter (Q, L, P)[8]. Here we consider an entity which starts with "Q" as a *normal entity*.
- If i_2 is a re-edit – **c/n**: Whether i_1 is the same as i_2, i.e., these are two consecutive edits (**c**) on the same entity or not (**n**).
  Otherwise (i_2 is a new edit) – **z/o/u**: Common classes (via the property *instance of*) between entity i_1 and i_2: **z** for zero classes in common, **o** for at least one class in common, and **u** for information is not available.
- **v/f/s**: Time difference between the two edits where **v** indicates *very fast* edit (less than three minutes), **f** refers to *fast* edit (less than 15 minutes), and **s** for *slow* edit (more than 15 minutes).

For example, **rncv** for a pair of entities (i_1, i_2) indicates that i_2 is a re-edit (r) and is a normal entity (n) which is the same as i_1 (c), and the time difference between the two consecutive edits is less than three minutes (v). Given this representation of edit history, pattern-based features can be extracted with the following two main steps [1].

First, frequent patterns from the *active* and *inactive* user edit histories in the training set are extracted separately using PrefixSpan [18] algorithm, which is one of the fastest sequential pattern mining algorithm. Those pattern-based features are extracted using SPFM data mining library [6]. Each pattern contains a sequence of pairs of entities consecutively edited by an editor where each pair is described using above-mentioned features (e.g., **rncv**). Afterwards, the frequency of each pattern f is calculated for both the *active* and *inactive* classes.

Next, two sets of patterns are extracted where the first one contains top frequent patterns that appear in both classes based on the absolute frequency difference between the two classes, and the second set contains top frequent patterns only appear for active class[9]. Finally, we select the set of top k patterns of length l that appear for both classes and for active editors only. We used the same setting as in [1] where $k = 13$ and $l \in \{1, 2, 3\}$, which results in a total of 78 Boolean features for any classifier as an input with 1 indicates a pattern appears in the edit history of an editor, 0 otherwise.

3.3 DeepFM as the Classification Model

On top of above-mentioned features, we adopt a DeepFM-based classification model which was proposed in [9] for recommender systems. To the best of our knowledge, this is the first attempt of utilizing the DeepFM model for user classification tasks on Wikidata. DeepFM was designed to automatically capture

[8] `https://www.wikidata.org/wiki/Wikidata:Identifiers`: items are prefixed with Q, properties are prefixed by P, lexemes are prefixed by L.

[9] Similar to the observation in [1], there is no pattern only appears for inactive editors.

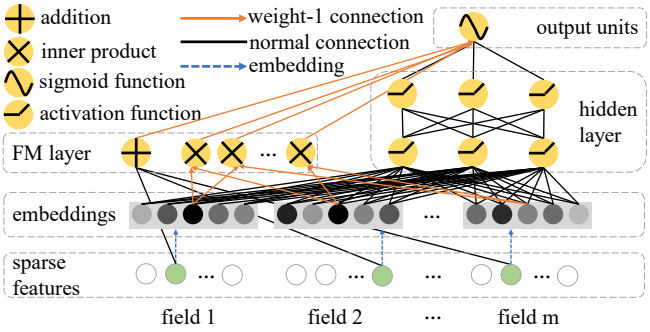


Fig. 1: DeepFM architecture from [9], which consists of two main components: FM part on the left and DNN part on the right.

feature interactions by modeling low-order feature interactions through Factorization Machines (FM) and high-order feature interactions via Deep Neural Networks (DNN). Intuitively, modeling different types of interactions via DeepFM plays a crucial role in capturing the temporal dynamics of editing behaviors.

The model consists of two main components: an FM component and DNN component as illustrated in Figure 1. Moreover, the model has the ability of memorizing low- and high-order feature interactions and generalizing feature combinations by jointly training the FM and DNN components for the combined prediction model:

$$\hat{y} = \sigma(\hat{y}_{FM} + \hat{y}_{DNN}) \tag{1}$$

where $\hat{y}_{FM}$ indicates the output from the FM part, $\hat{y}_{DNN}$ indicates the output from the DNN part of DeepFM, and σ is the *sigmoid* function. In the following, we briefly introduce the FM component and Deep component separately.

The *FM component* of DeepFM is a factorization machine introduced by Rendle et al. [22] to learn feature interactions for recommender systems. It consists of two parts with the first one considers linear interactions among features, and the second one with pairwise feature interactions as inner product of respective feature latent vectors as follows.

$$\hat{y}_{FM} = \langle \mathbf{w}, \mathbf{x} \rangle + \sum_{i=1}^{d} \sum_{j=i+1}^{d} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j \tag{2}$$

where $\mathbf{w}, \mathbf{x} \in \mathbb{R}^d$, $\mathbf{v}_i \in \mathbb{R}^k$, and d and k indicate the number of features and the dimension of a feature's latent factor, respectively. And $\langle \cdot, \cdot \rangle$ is the dot product of two vectors.

The *DNN component* is a feed-forward neural network which aims to learn high-order feature interactions. The first layer $\mathbf{a}^{(0)}$ in Eq. 3 is an embedding layer which compresses the input field vectors to the embedding vectors: $[\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_m}]$ for sparse/categorical features where for dense/numerical features the embedding layer will be ignored. DeepFM reuses the latent feature

vectors in the FM component as network weights which are learned and used for this compression, and the FM and DNN components share the same feature embeddings.

$$\mathbf{a}^{(0)} = [\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_m}], \;\; \mathbf{a}^{(l+1)} = \sigma(\mathbf{W}^{(l)}\mathbf{a}^{(l)} + \mathbf{b}^{(l)}) \tag{3}$$

$$\hat{y}_{DNN} = \mathbf{W}^{|H|+1}\mathbf{a}^{|H|} + \mathbf{b}^{|H|+1} \tag{4}$$

where $\mathbf{a}^{(l)}$, $\mathbf{W}^{(l)}$, and $\mathbf{b}^{(l)}$ refer to the output, weights, and bias of l-th layer, and $|H|$ is the total number of hidden layers.

Training details We use the cross-entropy loss (or log loss) as our loss function, and the objective is to minimize the loss over all N training examples.

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i)] \tag{5}$$

where y_i and $\hat{y}_i$ denote the ground truth and the predicted probability of class 1 for i-th instance, respectively. To resolve the overfitting problem, we use 20% of the training data as our validation set to adopt an early stopping strategy. We run up to 1,000 epochs for training, but the early stopping strategy stops the training if there is no improvement of *AUROC* (Area Under the Receiver Operating Characteristics) on the validation set.

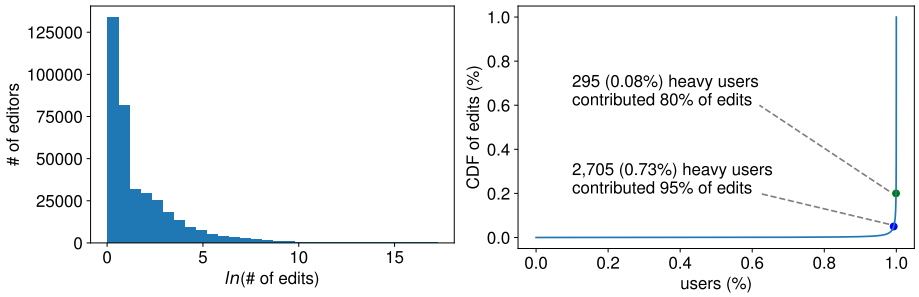
4 Wikidata Dataset

In this section, we discuss the Wikidata dataset for our study and provide exploratory analysis with respect to edit history of Wikidata users in Section 4.1. We use the Wikidata dump of 2020-12-01[10] with respect to edit history for our study. As we are interested in ordinary human editors who are registered on Wikidata, we further excluded edits from anonymous users (who we only know an IP address), bot accounts and administrators of Wikidata based on the open bot list[11] and admin list[12]. After filtering, the dataset contains 371,068 users with 519,121,793 edits in total for our analysis and experiments. The raw data includes crucial edit information for each edit such as:

- *Username* which indicates unique username who made an edit.
- *Time* with respect to the edit.
- *Entity* which refers to the unique entity ID such as `Q13580495`.
- *Edit action type* which denotes an automatic comment such as `wbcreateredirect` (Creates entity redirects) generated by the Wikidata's backend[13]. We choose top 50 edit action types which covers 99.9% of the whole data and treat the rest as "others".

We explore the filtered Wikidata dataset in detail in the following.

[10] `https://dumps.wikimedia.org/wikidatawiki/20201201/`
[11] `https://www.wikidata.org/wiki/Wikidata:Bots`
[12] `https://www.wikidata.org/wiki/Wikidata:Administrators`
[13] Registered actions in Wikidata's backend: `https://bit.ly/39NsrMh`



(a) (b)

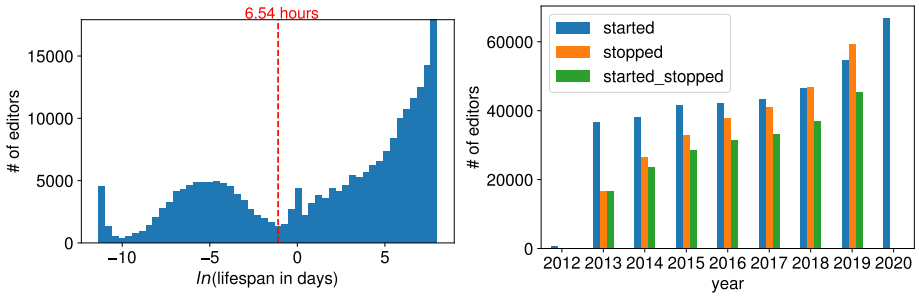
Fig. 2: (a) Histogram of editors in terms of the number of edits, and (b) CDF of edits for those editors on Wikidata until 2020-12-01.

4.1 Exploratory Analysis of the Dataset

First, we look at the distribution of edit volume on Wikidata. Figure 2a shows the histogram of the number of edits of all users. The figure illustrates that there are a lot of users making a low number of edits while a small number of *heavy users* making a high number of edits. The CDF (Cumulative Distribution Function) plot of edits in Figure 2b further illustrates this phenomenon. More specifically, the green dot in the figure refers to a point that 295 out of 371,068 users (0.08%) contributed 80% of the total edits. In addition, the blue dot indicates that 2,705 (0.73%) users contributed 95% of all edits, which again shows that a small number of users have contributed the majority of the edits.

Next, we look at the lifespan of those editors, i.e., the duration between a user's first and last edits in days. The histogram of Wikidata users' lifespan is shown in Figure 3a, where the lifespan of each user is scaled logarithmically using natural logarithm. As we can see from the figure, there are three distinct areas where the first two areas on the left correspond to occasional users who stopped editing Wikidata entities after the first few edits or newly joined editors, while the right area corresponds to editors with deeper interests in staying in the community and keeping editing Wikidata entities until they lose interest because of some reason. The separation point between the first two areas and the last one is 6.54 hours. This is shorter than the observation on editors' lifespan on Wikipedia [31] where the point for separation is around 8 hours.

Finally, we analyze the number of users who started or stopped editing Wikidata as well as that of users who started and stopped editing for each year from 2012 to 2020. On top of the same analysis done for the period between 2012 and 2016 in [24], our analysis including recent years could provide some insights on the trends the number of editors who started or stopped. Figure 3b shows an increasing trend over the years regarding the number of users in three aspects (i.e., who started, stopped, started and stopped). The numbers of stopped (as well as started and stopped) for 2020 are excluded in the figure as we have no



(a) (b)

Fig. 3: (a) Histogram of editors in terms of their lifespan in days, and (b) the number of editors started and stopped using Wikidata from 2012 to 2020.

clue about the last edits of users in 2020 are indeed their last edits. Hence, we cannot consider them as stopped. The figure illustrates that although the number of newcomers is increasing, the number of users who stopped contributing to Wikidata is also increasing, which again shows the importance of predicting leaving editors and additional efforts might be needed to keep those users, e.g., recommending entities of their interests to edit.

5 Experimental Setup

In this section, we discuss our experimental setup including datasets for training and testing our user classification approaches (Section 5.1), a set of methods for comparison and evaluation (Section 6.1), and evaluation metrics (Section 5.3).

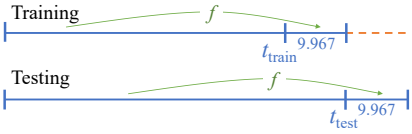
Instead of defining a new threshold for determining an editor as inactive based on his/her activity, we adopt the definition of a recent study from Sarasua et al. [24] for deriving the ground truth labels of editors. According to [24], if a user has not been editing any entity for 9.967 months (299 days), we consider the user as an *inactive* user (who stopped using Wikidata)[14].

5.1 Dataset

Similar to previous studies, we further limit users with more than one edit for training and testing different approaches including ours. Figure 4 illustrates how the dataset is divided into training and testing sets, which aims to resemble a real-world scenario. In the figure, t_{test} indicates the timestamp which is 9.967

[14] The threshold should make the majority of user labels (active or inactive) stable, e.g., the majority of inactive users decided based on their activities until a timestamp t and the predefined threshold should remain inactive after t. This suggests a higher value for the threshold is desirable, e.g., 9.967 months is better than 3 months, and we observe that the majority of inactive users do not visit the platform afterwards.

Fig. 4: Dataset split for training and testing and their statistics.



Training (# of users)			Test (# of users)		
Active	Inactive	Total	Active	Inactive	Total
29,509	31,283	60,792	32,068	33,500	65,568

months before the last edit time in our dataset, and t_{train} refers to the timestamp which is 9.967 months before t_{test}. We train a classification model f based on the edit history of users before t_{test} and their ground truth labels (i.e., inactive if there is no edit in the 9.967 months between t_{train} and t_{test}, and active otherwise), and test with f for predicting whether a user will be active or inactive after t_{test}. For both training and testing, we limit users who are *active* before t_{train} and t_{test} to predict whether those active users will remain active or become inactive. Take testing as an example, we limit active users based on their activity between t_{train} and t_{test}, i.e., those inactive users during the time between t_{train} and t_{test} by our definition are already "gone" and therefore excluded for testing. As summarized in Figure 4, there are 60,792 editors with 29,509 active and 31,283 inactive ones in the training set, and 65,568 editors in total with 32,068 active and 33,500 inactive ones in the test set for evaluation.

5.2 Compared Methods

Due to the difference between our task and the tasks of previous studies discussed in Section 2, and the difference between the Wikidata and Wikipedia datasets, those approaches from previous works could not apply to our problem directly. Nevertheless, we tried our best to adapt five previous approaches proposed in the context of Wikipedia or different tasks for our task in the context of Wikidata in addition to using DeepFM to investigate the classification performance. We use a naming convention of [`ML model`]-[`the first two characters of the main author`] in the following to distinguish those methods adapted for comparison, and provide their details.

`GBT-Zh` [31] uses Gradient Boosted Trees with three features such as the number of edits, the number of edited entities, and the length between the first and the last edit extracted from each of the 10 periods in P for each user in the same manner as ours. We tuned the max depth hyper-parameter with a grid search over $\{2, 4, 6, 8, 10\}$ using a 3-fold cross validation. The *drift* feature which measures the average number of edits of all editors in [31] is the same for all examples and is not important for our classification task although it is important for predicting the number of edits of users by capturing the overall changes of edit volume over time. Therefore, this feature is excluded in this adapted method.

`kNN-Zh` [31] leverages the same set of features as `GBT-Zh` but uses kNN as the classification model. The number of neighbors k is drawn from 200 to 3,000 in step of 200 using a 3-fold cross validation.

`RF-Sa` [24] is adapted to our context using Random Forest with a set of features such as the number of total edits, the average number of edits per item, the number of distinct items edited, and the diversity of types of edits extracted from each of the last 10 consecutive months introduced in [24]. Similar to `GBT-Zh`, we tuned the max depth hyper-parameter with a grid search over $\{2, 4, 6, 8, 10\}$ using a 3-fold cross validation. The ML model used in [24] was designed to analyze the edit volume of each user during his/her lifetime on Wikidata before he/she becomes inactive (or leaves the platform), and cannot be directly applied in our context for several reasons. For example, it trains a RANSAC model [5] for *each* user with the aforementioned features, and uses the parameters of the RANSAC model for training a Random Forest model to classify high/low volume or long/short lifespan editors with the focus on "gone" users. As one might expect, this results in not only training a great number of RANSAC models but also requires a good number of edits for each user for fitting the corresponding RANSAC model. Therefore, we use the set of features with Random Forest directly here as our baseline.

`LR-Sa` [24] leverages the same set of features as `RF-Sa` but uses Logistic Regression for classification. We tuned the regularization strength with a grid search over $\{0.1, 1, 10, 100, 1000\}$ using a 3-fold cross validation.

`SVM-Ar` is adapted from [1] which utilizes the set of pattern-based features introduced in Section 3.2 with a Support Vector Machine (SVM) classifier for predicting inactive editors. We tuned the regularization strength and the kernel parameter (γ) of a non-linear Radial Basis Function (RBF) kernel of SVM with a grid search over $\{0.1, 1, 10, 100, 1000\}$ and $\{1, 0.1, 0.01, 0.001, 0.0001\}$ using a 3-fold cross validation.

`DeepFM-Stat` refers to the proposed approach using DeepFM with the set of statistical features introduced in Section 3.1. We implemented `DeepFM-Stat` with DeepCTR library [26]. After tuning parameters such as dropout, batch sizes, and the number of hidden layers using 20% of the training set for validation, we opt to use the default DeepFM architecture of DeepCTR, which uses two hidden layers both with 128 nodes for its DNN part, embedding size $k = 4$, without dropout [27] as there is no significant improvement compared to the default architecture.

`DeepFM-Pattern` uses the same architecture as `DeepFM-Stat` but uses the set of pattern-based features used for `SVM-Ar` and introduced in Section 3.2.

`DeepFM-Stat+Pattern` considers both the statistical features and pattern-based features, which has not been explored together before. We investigate the synergistic effect of those two sets of features in the context of DeepFM.

We implemented `GBT-Zh`, `kNN-Zh`, `RF-Sa`, `LR-Sa`, and `SVM-Ar` with scikit-learn [17]. For `DeepFM` with different sets of features such as `Stat`, we run five times and the results in Section 6 are based on the averages over the five runs. All experiments are run on a laptop with Intel(R) Core(TM) i5-3230M (@2.6GHZ) processor and 8GB RAM.

5.3 Evaluation Metrics

We use *AUROC*/*AUC* (Area Under the Receiver Operating Characteristics), *F1* (F1 score) to evaluate the classification performance of different methods introduced in Section 6.1. The *F1* score: $F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$ is the harmonic mean of precision and recall, where the precision is the number of true positive predictions divided by the number of total positive predictions, and the recall refers to the number of true positive predictions divided by the number of all instances should have been predicted as positive. Compared to $F1$ where classification is based on a single threshold 0.5 (one if the predicted probability is equal or higher than the threshold, and zero otherwise), *AUROC* reflects the classification performance at various threshold settings by measuring the area under the ROC curve, which shows a trade-off between the true positive rate and false positive rate. We analyze the *F1* score of *inactive* class and *AUROC* as our focus here is predicting editors that would become inactive.

6 Results

In this section, we first investigate the classification performance of those methods introduced in Section 6.1, and analyze the usefulness of our statistical features introduced in Section 3.1 in detail.

6.1 Comparison with the Set of Different Methods

Here we discuss the results with respect to two questions mentioned in Section 1 such as what types of ML approaches perform well for the prediction, and how well those approaches applied to Wikipedia can perform in the context of Wikidata. Table 1 shows the overall classification results in terms of *AUROC* and *F1* with the set of methods compared. As shown in the table, for approaches using either *statistical* or *pattern-based* features, `DeepFM-Stat` achieves the best performance of *AUROC* (0.8928) and *F1* (0.8247) followed by `GBT-Zh`. Despite the fact that `GBT-Zh` is adapted from edit volume prediction on Wikipedia, we notice that the approach provides a strong performance with an *AUROC* score of 0.8890 and an *F1* score of 0.8205 for classifying inactive users on Wikidata as well. Similar to the observation for predicting edit volume on Wikipedia, the classification performance of `kNN-Zh` is worse than `GBT-Zh` with the same set of features. `RF-Sa` and `LR-Sa`, which are adapted from the method for analyzing the edit volume of a Wikidata editor during his/her lifetime on the platform, perform worse than other methods. Overall, `DeepFM-Stat` improves the *AUROC* score 0.43%-16.75% and the *F1* score 0.51%-6.15% compared to those non-DeepFM alternatives either using statistical features or pattern-based ones for classifying inactive editors on Wikidata.

Next, we investigate what types of features (e.g., statistical, pattern-based, or both of them) are useful in the context of DeepFM. We observe

Table 1: Classification performance of the set of methods compared in terms of *AUROC* and *F1* with the best-performing ones in **bold**.

Method	*AUROC*	*F1*
GBT-Zh	0.8890	0.8205
kNN-Zh	0.8731	0.7935
RF-Sa	0.7647	0.7769
LR-Sa	0.7656	0.7795
SVM-Ar	0.8396	0.8029
DeepFM-Pattern	0.8786±0.0002	0.7992±0.0028
DeepFM-Stat	0.8928±0.0001	0.8247±0.0006
DeepFM-Stat+Pattern	**0.9561±0.0005**	**0.8843±0.0012**

that DeepFM-Stat+Pattern, which considers both the statistical and pattern-based features together, further improves the performance significantly compared to using either statistical or pattern-based features alone. For example, DeepFM-Stat+Pattern achieves an *AUROC* score of 0.9561 and an *F1* score of 0.8843, which outperforms DeepFM-Stat (+7.09% of *AUROC*, +7.23% of *F1*) and DeepFM-Pattern (+8.82% of *AUROC*, +10.65% of *F1*). This shows that the two types of features – statistical and pattern-based ones – can complement each other and achieves the best classification performance on predicting inactive editors, which has not been explored in previous studies[15].

Finally, Figure 5 illustrates the time required for training each of those methods where we observe all except SVM-Ar can be trained within 30 minutes. As one might expect, leveraging both statistical and pattern-based features (DeepFM-Stat+Pattern) results in increased training time compared to only using either of the sets of features. The training/fitting time for all methods is arguably reasonable, which can be updated periodically (e.g., every day or week) easily, and the time can be further improved with a better infrastructure.

6.2 Analysis of Statistical Features

In this section, we analyze the set of statistical features introduced in Section 3.1 to answer the following questions: (1) *Are those features improving the classification performance compared to the features adapted in previous studies when the same ML approach is applied?* (2) *How much improvement can we achieve by considering a greater number of periods in P?* (3) *Which statistical feature contributes the most to the performance with DeepFM?*

Contribution of proposed features. Here we investigate whether using our features improves the classification performance compared to using the set of features that are adapted from previous studies when the same ML approach is applied. Table 2 illustrates the performance in the context of four different

[15] Similar trends can be observed when we limit users with at least five edits instead of one. For example, DeepFM-Stat+Pattern provides the best performance and improves *AUROC* 10.7% and 11.5% compared to using GBT-Zh and kNN-Zh.

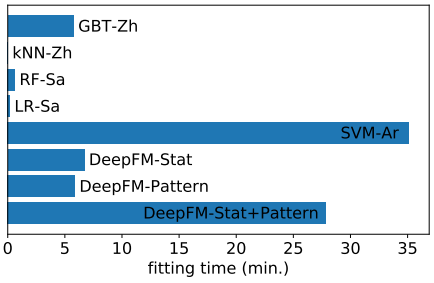


Fig. 5: Time required for training.

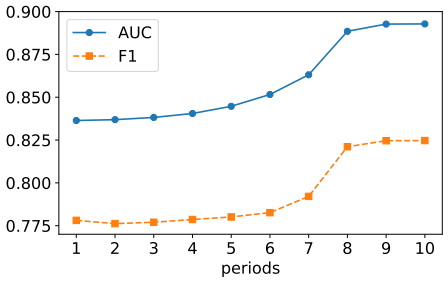


Fig. 6: Impact of the number of periods.

Table 2: Proposed features used with different classification approaches.`*-Stat` indicates `*` model with our statistical features introduced in Section 3.1.

	AUROC (Improvement)	*F1* (Improvement)
`GBT-Zh`	0.8890	0.8205
`GBT-Stat`	**0.8918** (+0.32%)	**0.8225** (+0.24%)
`kNN-Zh`	0.8731	0.7935
`kNN-Stat`	**0.8898** (+1.91%)	**0.8172** (+2.99%)
`SVM-Ar`	0.8396	0.8029
`SVM-Stat`	**0.8640** (+2.91%)	**0.8040** (+0.14%)
`DeepFM-Zh`	0.8922±0.0001	0.8223±0.0029
`DeepFM-Stat`	**0.8928±0.0001** (+0.07%)	**0.8247±0.0006** (+0.29%)

methods such as GBT, kNN, SVM, and DeepFM. The results show that using our statistical features consistently achieves better *AUROC* and *F1* scores with those methods, which indicates the effectiveness of those features.

Contribution of periods in P. Next, we demonstrate how the performance changes with the increasing number of periods considered to construct the set of statistical features. More specifically, we start from considering the shortest period ($\frac{1}{16}$ for p), and add the next period in P one by one for each experiment. The experimental results are shown in Figure 6. It illustrates that taking more periods for extracting temporal dynamic features improves the performance for both *AUROC* and *F1* where the improvement gradually diminishes. In particular, the performance is improved significantly when the 8th period (12 months) is added where the *AUROC* improves from 0.8631 to 0.8885 (+2.94%), and the *F1* score improves from 0.7921 to 0.8211 (+3.66%).

Contribution of each feature. Finally, we investigate which feature contributes the most to the classification performance by removing each feature. The results are illustrated in Figure 7 where the performance is averaged over five runs when each feature is removed. The figure demonstrates that, $N_{total \cdot edit \cdot ent}$ – the number of total edits in the last $p \in P$ months – is a set of features that contribute the most in terms of both *AUROC* and *F1*. This is in line with the findings from [7] which shows the number of edits before is a good indicator

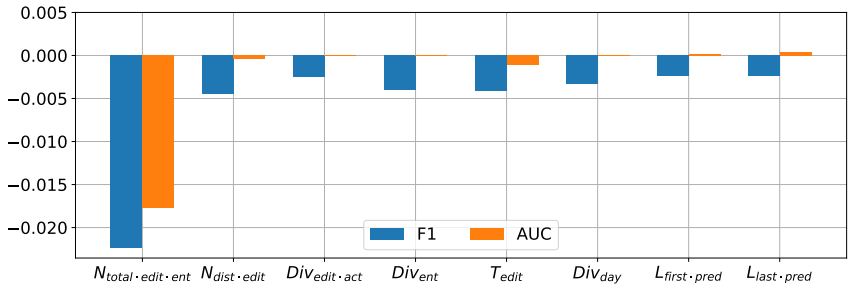


Fig. 7: Impact on `DeepFM-Stat` when each statistical feature is removed.

of the future edits of an editor on Wikipedia. We also notice that all other features contribute to a certain extent for *F1* score while the contribution of other features except $N_{total \cdot edit \cdot ent}$ to $AUROC$ is lower than that to *F1*.

7 Conclusions

In this paper, we investigated a wide variety of classification approaches to predict Wikidata editors who will leave the platform. To the best of our knowledge, this is the first work on predicting inactive editors on Wikidata. To this end, we investigated a set of proposed statistical features together with a set of adapted pattern-based features and leveraged DeepFM as the classification method from recommender systems. In addition, we also built several adapted approaches from previous studies in the context of Wikipedia or different tasks, and investigated those approaches for our classification problem regarding Wikidata editors. We have shown, in the evaluation, that `DeepFM-Stat` achieves the best $AUROC$ and *F1* performance compared to other adapted methods when using either set of features. In addition, we also showed that using both statistical and pattern-based features can improve the performance significantly and achieves the best $AUROC$ and *F1* score. The promising results with DeepFM indicate that other alternative models invented for recommender systems [12,30] can be explored in the future for our problem. We also believe our Wikidata dataset containing the edit history over seven years will benefit the research community for studying other aspects related to Wikidata such as recommender systems for recommending entities of interest or understanding edit behavior difference between registered and anonymous editors etc.

Despite the promising performance, there are many interesting questions left unanswered. First, our work does not reveal the set of most influential patterns of the 78 patterns which might provide some interesting insights regarding important edit patterns for the prediction. Secondly, we adopted the threshold – 9.967 months – from a recent study [24] for determining whether an editor left the platform. The impact of using different threshold values for deriving the

ground truth labels of editors needs to be further investigated. Finally, current approaches require manual feature engineering. In the future, we plan to focus on end-to-end approaches and investigate whether an end-to-end classification model without manual feature engineering is feasible.

Acknowledgements We thank the reviewers for the helpful feedback. Weipeng Huang is supported by SFI under the grant SFI/12/RC/2289_P2.

References

1. Arelli, H., Spezzano, F.: Who will stop contributingf predicting inactive editors in wikipedia. In: 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). pp. 355–358. IEEE (2017)
2. Arelli, H., Spezzano, F., Shrestha, A.: Editing behavior analysis for predicting active and inactive users in wikipedia. In: IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. pp. 127–147. Springer (2018)
3. Cuong, T.T., Müller-Birn, C.: Applicability of sequence analysis methods in analyzing peer-production systems: a case study in wikidata. In: International Conference on Social Informatics. pp. 142–156. Springer (2016)
4. Druck, G., Miklau, G., McCallum, A.: Learning to predict the quality of contributions to wikipedia. WikiAI **8**, 7–12 (2008)
5. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)
6. Fournier-Viger, P., Lin, J.C.W., Gomariz, A., Gueniche, T., Soltani, A., Deng, Z., Lam, H.T.: The spmf open-source data mining library version 2. In: ECML/PKDD. pp. 36–40. Springer (2016)
7. Gandica, Y., Carvalho, J., Dos Aidos, F.S.: Wikipedia editing dynamics. Physical Review E **91**(1), 012824 (2015)
8. Geiger, R.S., Halfaker, A.: Using edit sessions to measure participation in wikipedia. In: CSCW. pp. 861–870 (2013)
9. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: A factorization-machine based neural network for ctr prediction. In: IJCAI (2017)
10. Kaffee, L.A., Elsahar, H., Vougiouklis, P., Gravier, C., Laforest, F., Hare, J., Simperl, E.: Mind the (language) gap: Generation of multilingual wikipedia summaries from wikidata for articleplaceholders. In: ESWC. pp. 319–334. Springer (2018)
11. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al.: Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web **6**(2), 167–195 (2015)
12. Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., Sun, G.: xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1754–1763 (2018)
13. Malyshev, S., Krötzsch, M., González, L., Gonsior, J., Bielefeldt, A.: Getting the most out of wikidata: Semantic technology usage in wikipedia's knowledge graph. In: International Semantic Web Conference. pp. 376–394. Springer (2018)

14. Mora-Cantallops, M., Sánchez-Alonso, S., García-Barriocanal, E.: A systematic literature review on wikidata. Data Technologies and Applications (2019)
15. Müller-Birn, C., Karran, B., Lehmann, J., Luczak-Rösch, M.: Peer-production system or collaborative ontology engineering effort: What is wikidata? In: Proceedings of the 11th International Symposium on Open Collaboration. pp. 1–10 (2015)
16. Panciera, K., Halfaker, A., Terveen, L.: Wikipedians are born, not made: a study of power editors on wikipedia. In: CSCW. pp. 51–60 (2009)
17. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. JMLR **12**, 2825–2830 (2011)
18. Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: Mining sequential patterns by pattern-growth: The prefixspan approach. IEEE Transactions on Knowledge and Data Engineering **16**(11), 1424–1440 (2004)
19. Pellissier Tanon, T., Kaffee, L.A.: Property label stability in wikidata: evolution and convergence of schemas in collaborative knowledge bases. In: Companion Proceedings of the The Web Conference 2018. pp. 1801–1803 (2018)
20. Piscopo, A., Phethean, C., Simperl, E.: What makes a good collaborative knowledge graph: group composition and quality in wikidata. In: International Conference on Social Informatics. pp. 305–322. Springer (2017)
21. Piscopo, A., Vougiouklis, P., Kaffee, L.A., Phethean, C., Hare, J., Simperl, E.: What do wikidata and wikipedia have in common? an analysis of their use of external references. In: Proceedings of the 13th International Symposium on Open Collaboration. pp. 1–10 (2017)
22. Rendle, S.: Factorization machines. In: 2010 IEEE International Conference on Data Mining. pp. 995–1000. IEEE (2010)
23. Ristoski, P., Paulheim, H.: Rdf2vec: Rdf graph embeddings for data mining. In: International Semantic Web Conference. pp. 498–514. Springer (2016)
24. Sarasua, C., Checco, A., Demartini, G., Difallah, D., Feldman, M., Pintscher, L.: The evolution of power and standard wikidata editors: comparing editing behavior over time to predict lifespan and volume of edits. CSCW **28**(5), 843–882 (2019)
25. Shannon, C.E.: A mathematical theory of communication. acm sigmobile mob. Comput. Commun. Rev **5**(1), 3–55 (2001)
26. Shen, W.: Deepctr: Easy-to-use,modular and extendible package of deep-learning based ctr models. `https://github.com/shenweichen/deepctr` (2017)
27. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research **15**(1), 1929–1958 (2014)
28. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)
29. Waagmeester, A., Stupp, G., Burgstaller-Muehlbacher, S., Good, B.M., Griffith, M., Griffith, O.L., Hanspers, K., Hermjakob, H., Hudson, T.S., Hybiske, K., et al.: Science forum: Wikidata as a knowledge graph for the life sciences. Elife **9** (2020)
30. Yang, Y., Xu, B., Shen, S., Shen, F., Zhao, J.: Operation-aware neural networks for user response prediction. Neural Networks **121**, 161–168 (2020)
31. Zhang, D., Prior, K., Levene, M., Mao, R., van Liere, D.: Leave or stay: The departure dynamics of wikipedia editors. In: International Conference on Advanced Data Mining and Applications. pp. 1–14. Springer (2012)