Empirical Software Engineering (2019) 24:3316–3355
https://doi.org/10.1007/s10664-019-09706-9

Towards understanding and detecting fake reviews in app stores

Daniel Martens[1] · **Walid Maalej**[1]

Published online: 10 May 2019

Abstract
App stores include an increasing amount of user feedback in form of app ratings and reviews. Research and recently also tool vendors have proposed analytics and data mining solutions to leverage this feedback to developers and analysts, e.g., for supporting release decisions. Research also showed that positive feedback improves apps' downloads and sales figures and thus their success. As a side effect, a market for fake, incentivized app reviews emerged with yet unclear consequences for developers, app users, and app store operators. This paper studies fake reviews, their providers, characteristics, and how well they can be automatically detected. We conducted disguised questionnaires with 43 fake review providers and studied their review policies to understand their strategies and offers. By comparing 60,000 fake reviews with 62 million reviews from the Apple App Store we found significant differences, e.g., between the corresponding apps, reviewers, rating distribution, and frequency. This inspired the development of a simple classifier to automatically detect fake reviews in app stores. On a labelled and imbalanced dataset including one-tenth of fake reviews, as reported in other domains, our classifier achieved a recall of 91% and an AUC/ROC value of 98%. We discuss our findings and their impact on software engineering, app users, and app store operators.

Keywords Fake reviews · App reviews · User feedback · App stores

1 Introduction

In app stores, users can rate downloaded apps on a scale from 1 to 5 stars and write a review message. Thereby, they can express satisfaction or dissatisfaction, report bugs, or suggest new features (Carreno and Winbladh 2013; Pagano and Maalej 2013; Maalej et al. 2016a). Similar to other online stores, before downloading an app, users often read through the

Communicated by: David Lo, Meiyappan Nagappan, Fabio Palomba and Sebastiano Panichella

✉ Daniel Martens
martens@informatik.uni-hamburg.de

✉ Walid Maalej
maalej@informatik.uni-hamburg.de

[1] Department of Informatics, University of Hamburg, Hamburg, Germany

reviews. Research found that ratings and reviews correlate with sales and download ranks (Harman et al. 2012; Pagano and Maalej 2013; Svedic 2015; Martin et al. 2016; Finkelstein et al. 2017). Stable numerous ratings lead to higher downloads and sales numbers.

As a side effect, an illegal market for fake app reviews has emerged, with the goal to offer services that help app vendors improve their ratings and ranking in app stores. According to app store operators, in regular app reviews, real users are supposed to be triggered by their satisfaction or dissatisfaction of using the app to provide feedback. Fake reviewers, however, get paid or similarly rewarded to submit reviews. They might or might not be real users of the app. Their review might or might not be correct and reflecting their opinion.

We refer to this type of non-spontaneous, requested, and rewarded reviews as **fake reviews**. Fake reviews are prohibited in popular app stores such as in Google Play (Google 2017) or Apple App Store (Apple 2017). For instance, Apple states: "*If we find that you have attempted to manipulate reviews, inflate your chart rankings with paid, incentivized, filtered, or fake feedback, or engage with third party services to do so on your behalf, we will take steps to preserve the integrity of the App Store, which may include expelling you from the Developer Program.*".

Recently, Google highlighted the negative effects of fake reviews in an official statement and explicitly asks developers to not buy and users to not accept payments to provide fake reviews (Google 2019). Even governmental competition authorities started taking actions against companies using fake reviews to embellish their apps. For instance, the Canadian telecommunication provider Bell was fined $1.25 million (9to5Mac 2017) for faking positive reviews to their apps. Vice versa, the CNN app was affected by thousands of negative fake reviews to decrease its rating and ranking within the Apple App Store (DigitalTrends 2018).

While the phenomena of fake participation (e.g., in form of commenting, reporting or reviewing) is well-known in domains such as online journalism (Lee et al. 2010; Ferrara et al. 2014; Dickerson et al. 2014; Subrahmanian et al. 2016) or on business and travel portals (Jindal and Liu 2008; Ott et al. 2011; Feng et al. 2012; Mukherjee et al. 2013b), it remains understudied in software engineering—in spite of recent significant research on app store analysis and feedback analytics (Martin et al. 2016).

Fake reviews threaten the integrity of app stores. If real users don't trust the reviews, they probably will refrain from reading and writing reviews themselves. This can result into a problem for app store operators, as app reviews is a central concept of the app store ecosystem. Fake reviews can have negative implications for app developers and analysts as well. Numerous software and requirements engineering researchers studied app reviews, e.g., to derive useful development information such as bug reports (Khalid 2013; Maalej et al. 2016a) or to understand and steer the dialogue between users and developers (Iacob and Harrison 2013; Oh et al. 2013; Johann et al. 2017; Villarroel et al. 2016). Further, researchers (Carreno and Winbladh 2013; Chen et al. 2014; Maalej et al. 2016b) and more recently tool vendors (AppAnnie 2017) suggested tools that derive actionable information for software teams from reviews such as release priorities and app feature co-existence. None of these works considers fake reviews and their implications. Negative fake reviews, e.g., by competitors reporting false issues, can lead to confusion and waste of developers' time. Positive fake reviews might also lead to wrong insights about real users needs and requirements.

In this paper, we study fake app reviews, focusing on three research questions:

RQ1 **How and by whom are app ratings and reviews manipulated?**
Through online research and an investigative disguised questionnaire, we identified 43 fake review providers and gathered information about their fake reviewing strategies and offers.

RQ2 **How do fake reviews differ from regular app reviews?**
We crawled ∼60,000 fake reviews, empirically analyzed and compared them with ∼62 million official app reviews from the Apple App Store. We report on quantitative differences of fake reviewers and concerned apps.

RQ3 **How accurate can fake reviews be automatically detected?**
We developed a supervised classifier to detect fake reviews. Within an in-the-wild experiment, we evaluated the performance of multiple classification algorithms, configurations, and classification features.

In Section 2 we introduce the research questions, method, and data. We then report on the results along the research questions: fake review market in Section 3, characteristics in Section 4, and automated detection in Section 5. Afterwards, we discuss the implications and limitations of our findings in Section 6. Finally, we survey related work in Section 7 and conclude the paper in Section 8.

2 Study Design

We first introduce our research questions. Then, we describe our research method and data along the data collection, preparation, and analysis phase.

2.1 Research Questions

We aim to qualitatively and quantitatively understand fake app reviews including their market, characteristics, and potential automated detection. In the following we detail our research questions by listing the sub-questions we aim to answer.

RQ1 **Fake review market** reveals how app sales and downloads are manipulated and to which conditions. We investigate the following questions:

1. *Providers:* By whom are fake reviews offered? What strategies do fake review providers follow?
2. *Offers:* What exact services do fake review providers offer and under which conditions?
3. *Policies:* What are providers policies for submitting fake reviews? Do these reveal indicators to detect fake reviews?

RQ2 **Fake review characteristics** reveal empirical differences between official and fake reviews, including reviewed apps and reviewers.

1. *Apps:* Which apps are typically affected by fake reviews? What are their categories, prices, and deletion ratio?
2. *Reviewers:* What is a typical fake reviewer, e.g., in terms of number of reviews provided and review frequency?
3. *Reviews:* How do official and fake reviews differ, e.g., with regard to rating, length, votes, submission date, and content?

RQ3 **Fake review detection** examines how well supervised machine learning algorithms can detect fake reviews. We focus on the following questions:

1. *Features:* Which machine learning features can be used to automatically detect fake reviews?

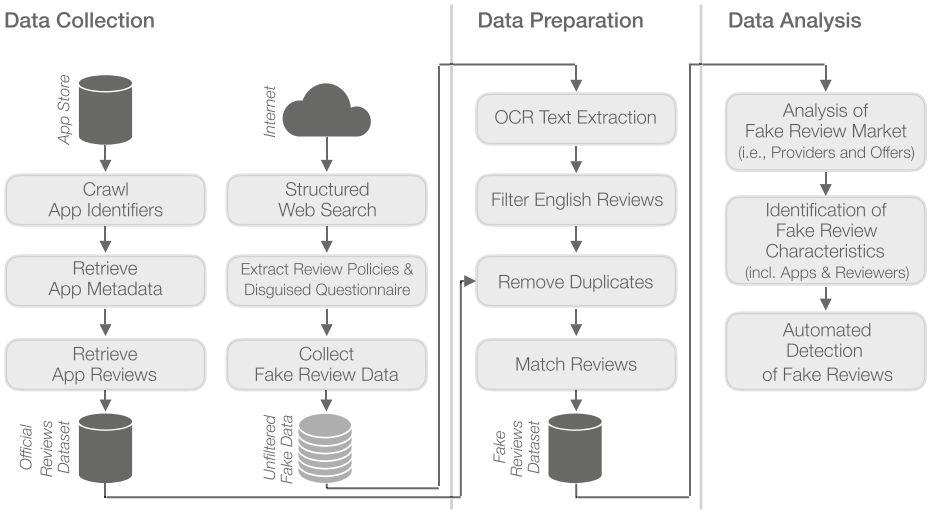

Fig. 1 Research method including data collection, preparation, and analysis phases

2. *Classification:* Which machine learning algorithms perform best to classify reviews as fake/non-fake?
3. *Optimization:* How can classifiers further be optimized? What is the relative importance of the classification features?
4. *In-the-Wild Experiment:* How do the classifiers perform in practice on imbalanced datasets with different proportional distributions of fake and regular reviews?

2.2 Research Method and Data

Our research method consists of a data collection, preparation, and analysis phase, as depicted in Fig. 1. We detail on each of the three phases in the following.

2.2.1 Data Collection Phase

For this study we collected two datasets: an official reviews dataset including app metadata and reviews from the Apple App Store; as well as a fake reviews dataset including metadata of apps affected by fake reviews, and fake reviews itself.

The **official reviews dataset** created in March 2017 consists of 1,430,091 apps, their metadata, and reviews. To collect the data, we implemented a distributed crawling tool using GoLang based on iTunes APIs, which we deployed on hundreds of cloud servers. The data collection included three steps. First, we crawled a list of all app identifiers available on the Apple App Store of the United States, as we focus on English reviews. Second, we obtained the metadata for each app, including the category, price, and number of reviews, using the iTunes Search API.[1] Last, we retrieved the app reviews using an internal iTunes API.

Overall, the Apple App Store included 207,782,199 ratings of which 67,727,914 (24.6%) have a review. We were able to crawl 62,617,037 (92.4%) of these reviews, as iTunes does not allow to receive more than 30,000 reviews per app. The size of the dataset is 36.58 GB.

[1] https://affiliate.itunes.apple.com/resources/documentation/itunes-store-web-service-search-api/

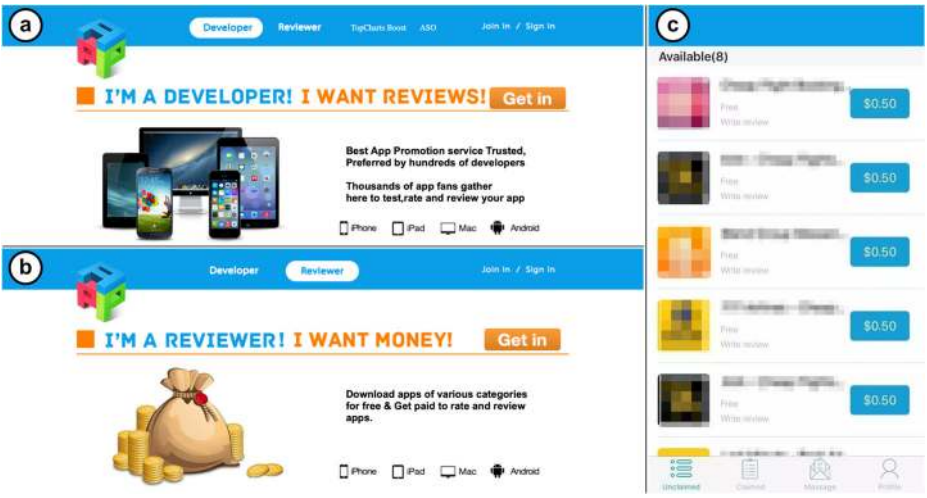

Fig. 2 Screenshot of a fake review provider, offering **a** app developers to buy reviews and **b** non-/developers to sign-up and write rewarded reviews. Section **c** shows a list of apps requesting fake reviews against monetary reward (obfuscated by the authors)

The crawled reviews were written by 25,333,786 distinct reviewers, i.e., users with different Apple IDs.[2] On average every reviewer submits around 2.47 reviews. The oldest app review was entered on 10/07/2008, therefore our dataset spans for nearly 9 years.

The **fake reviews dataset** was collected in April 2017 following three steps. First, we identified 43 fake review providers by performing a structured manual Google web search. To identify relevant search terms, we initially searched for the phrase "buy app reviews". We extracted related search terms suggested by the search engine. For those we repeated the previous step, resulting in 39 unique search terms, which are included in the replication package. Afterwards, we crawled the results of the ten first pages for each search term. We removed duplicate results and marked each result as fake review provider, relevant discussion about fake reviews (e.g., in forums), or irrelevant result. From relevant discussions we extracted additional fake review providers by reading through all sub-pages of the discussions. Then, we manually extracted the provider's offers from their websites.

In the second step, we conducted a disguised questionnaire to collect initial indicators for fake reviews such as the minimum star-rating and length. The questionnaire was presented as a request for buying fake reviews and sent to all providers per email on 26/04/2017.[3] For providers offering users to sign-up as fake reviewers to exchange or get paid for providing fake reviews, we created accounts and extracted the policies submitted fake reviews must comply with (cf. Fig. 2).

In the third and last step, we collected the fake review data, i.e., lists of apps requesting fake reviews and fake reviews itself. For this, we used three approaches:

1. *Social investigation*, i.e., asking the providers for fake review examples, while pretending to be interested in their services. We contacted the providers via email or live-chats

[2] https://appleid.apple.com/faq/#!&page=faq

[3] Conducted with permission of the Ethics Committee of the University of Hamburg.

Table 1 Overview of collected fake reviews and apps affected per provider (extracted using the listed approaches, dashes indicate that no apps or reviews could be extracted)

Provider Id	Provider Type	# Apps	# Reviews	Approach
PRP10	Paid Review Provider	77	–	Crawl
PRP16	Paid Review Provider	19	4	Crawl, Social
PRP21	Paid Review Provider	3	–	Social
PRP25	Paid Review Provider	–	3	Social
PRP26	Paid Review Provider	–	10	Social
PRP28	Paid Review Provider	–	3	Social
REP1	Review Exchange Portal	268	–	Crawl
REP2	Review Exchange Portal	277	–	Crawl
REP3	Review Exchange Portal	2,007	60,411	API, Crawl
REP5	Review Exchange Portal	7	–	Crawl
REP6	Review Exchange Portal	9	–	Crawl
REP8	Review Exchange Portal	182	–	Crawl
REP9	Review Exchange Portal	4	–	Crawl
		$\sum = 2,853$	$\sum = 60,431$	

on their websites. Using this strategy we received 3 apps and 20 fake reviews from 5 providers.

2. *Crawling*, for providers offering to sign-up as fake reviewers, we checked if the lists of apps requesting fake reviews are available (see Fig. 2, part c). To extract the apps we implemented crawlers. A sample crawler is included in the replication package. Overall, we collected 2,850 apps from 9 providers.
3. *APIs*, we found that providers require reviewers to upload screenshots of their reviews as a proof. We searched for publicly accessible screenshots and downloaded them. Based on this, we gathered 60,411 reviews from a single provider.

Overall we identified 2,853 apps and 60,431 reviews from 13 providers, see Table 1. Per provider the number of extracted apps and reviews is given, in case we could successfully apply at least one of the introduced approaches. The size of the collected data is 11.29 GB. We refer to this as **unfiltered fake data** (cf. Fig. 1), as it needs to be prepared for further analysis. For example, reviews within the dataset could have already been removed from the app store. For data preparation and analysis, all data, except the screenshots, is persisted as Parquet files and analyzed with Apache Zeppelin and Spark.[4]

2.2.2 Data Preparation Phase

Most fake reviews were collected in form of screenshots as shown in Fig. 3. We converted the screenshots into text using the Tesseract OCR engine.[5] We removed incomplete reviews that do not include a full readable title and body, e.g., if the title was outside the screen's visible area. Then, we used the Language Identification (LangID) library[6] to retrieve fake

[4] https://zeppelin.apache.org/
[5] https://github.com/tesseract-ocr/tesseract
[6] https://github.com/saffsd/langid.py

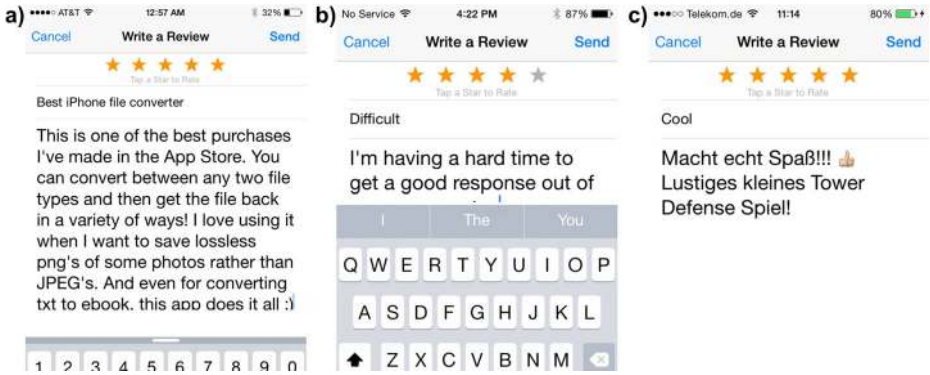

Fig. 3 Screenshots of fake reviews before submission to the app store used as proof for fake review providers, depicting **a** fake review included in our study, **b** cut-off fake review excluded from the study, and **c** non-English fake review also excluded from the study

reviews in English language only. We removed 7,445 reviews (12.32%) resulting in 52,986 fake reviews in English language.

Since the screenshots show the review edit screens before submitting the reviews to the app store, we further filtered the collected fake reviews for three possible reasons. First, we cannot assure that the reviews were actually submitted. Second, the reviews could have not been unlocked by the app store operator. Third, the reviews could have been deleted. Therefore, we only considered fake reviews that have been published to and still exist within the Apple App Store, i.e., which we could identify in the official reviews dataset as well.

For uniquely identifying (i.e., matching) reviews from the fake reviews dataset within the official reviews dataset, we removed duplicate reviews which consist of the same title and body within both datasets. Thereby, we removed 4,298 (8.11%) fake reviews leaving 48,688 items. The percentage of duplicate reviews within the official reviews dataset is with 16.08% nearly twice as high, which may be an indicator for the high diversity of fake reviews. We performed the matching using exact text comparison and by comparing the reviews' Levensthein distances. We used the Levensthein distance as single characters on the screenshots were sometimes not parsed correctly by the OCR engine. We searched for all fake reviews within the official reviews dataset by using an edit distance of up to 10 characters. For possible matches identified using the Levensthein distance, we manually verified if one of the suggested pairs is a match using two human annotators comparing the screenshot of the fake review and the possible matches. In case of disagreements (3% of all cases), a third annotator resolved the conflict. We matched 6,020 reviews by exact text comparison and 2,584 reviews by comparing the Levensthein distance.

Overall, we were able to identify 8,607 of the 60,431 (14.2%) collected fake reviews within the Apple App Store. These reviews were extracted from 5 providers. We also matched *apps* affected by fake reviews against the official reviews dataset, as the apps might not be available in the US App Store or might have been deleted. Of the 2,853 collected apps we found 2,174 apps (76.2%) in the official reviews dataset. Further, we identified 898 additional apps by extracting the app identifiers from previously matched fake reviews, resulting in 3,072 apps. We removed all apps that did not receive reviews within the app store, resulting in 1,929 of 3,072 (62.8%) apps provided by 10 different providers. Finally, we identified 721 fake *reviewers*, i.e., accounts of persons submitting fake reviews to the app store, by extracting their user identifiers from fake reviews.

Table 2 Overview of the official and the fake reviews datasets

	Official Reviews Dataset	Fake Reviews Dataset
# of reviews	62,617,037	8,607
# of apps	1,430,091	1,929
# of reviewers	25,333,786	721

In summary, after data cleaning the **fake reviews dataset** consists of 43 providers and structural information about their offers and policies, as well as **8,607 fake reviews, 1,929 apps affected by fake reviews, and 721 fake reviewers**. The dataset spans for nearly 7 years, as the oldest fake review was entered on 16/10/2010. Table 2 summarizes the official and fake reviews datasets.

2.2.3 Data Analysis Phase

The data analysis phase consists of three steps which respectively answer the research questions. To answer our first research question regarding the fake review **market**, we analyzed the qualitative data aggregated from the providers' websites, collected during the questionnaire, and extracted from the review policies.

To explore the fake review **characteristics** we applied a statistical analysis of the reviews, apps, and reviewers. We compare the figures from the fake reviews dataset to those from the official reviews dataset and run statistical tests whenever applicable. For example, we found that most fake reviews are provided for games. While regular apps receive most reviews on the day of an app release, apps affected by fake reviews receive most reviews eleven days after the update. This could indicate that apps affected by fake reviews do not have a real users basis that intrinsically provides reviews in reaction to changes introduced by app updates. Further, we found that fake reviewers provide twelve times as much reviews, with a four times higher frequency. Also, fake reviews have more positive ratings compared to official reviews, however the biggest difference exists between the amount of 1-star ratings.

To **detect** fake reviews, we created a labeled and balanced dataset of fake reviews and official reviews and used it to train and evaluate multiple classifiers, based on machine learning features that we derived from the analysis of characteristics. We conducted a hyperparameter tuning of the classifiers and evaluated the importance of the classification features. Finally, we performed an in-the-wild experiment to get more realistic results of how the classifiers perform in practice. Therefore, we used imbalanced datasets of fake and regular reviews. We varied the skewness of the datasets between 90% to 0.1% fake reviews and compared the classification results.

We detail each of these analysis steps in the following chapters. To support replication, our dataset and the analyses source code as Zeppelin notebooks are publicly available on our website.[7]

3 Fake Review Market (RQ1)

This section describes fake review providers and their market strategies, as well as offers and pricing models. Afterwards, pretended characteristics of fake reviews are summarized based on the results of the disguised questionnaire and analysis of reviewing policies.

[7] https://mast.informatik.uni-hamburg.de/app-review-analysis/

3.1 Review Providers and Market Strategies

We identified 43 providers offering fake reviews. These can be separated into two groups by their strategies used to supply reviews.

Paid review providers (PRP) accept payments to provide fake reviews. This applies for 34 out of 43 (79%) providers. User can select a package of, e.g., 50 reviews, specify their app name and identifier, and purchase it via Paypal or similar services. Afterwards, the fake reviews are submitted to the app store.

Review exchange portals (REP) allow app developers to sign-up and exchange reviews. The remaining 9 providers (21%) belong to this group. After sign-up developers browse through a list of apps requesting fake reviews. Figure 4 shows a sample request for fake reviews. Depending on their policies, review exchange portals ask users to submit fake reviews, e.g., with predefined ratings and review messages. For each fake review the developer submits, one credit is given as a reward. Developers with at least one credit can add their app to the list. Then, the credits are redeemed into reviews written by other developers.

In some cases, review exchange portals allow developers to buy credits and non-developers to sign-up and submit fake reviews. Non-developers are rewarded using micro-payments, typically between $0.20 to $1.50 per fake review.

Figure 5 shows the strategies of the fake review providers. After deciding to buy fake reviews at a paid review provider or to exchange (or buy) reviews at a review exchange portal, the developer provides basic information, such as the application identifier and whether the reviews should be positive or negative. Optionally, further information, such as keywords to be included within the reviews or predefined review messages, can be submitted. Using this information, the provider creates a review request (see, e.g., Fig. 4).

Review exchange portals publish these requests on their internal platform to recruit fake reviewers. For paid review providers the publishing process is not transparent. Using social investigation and by offering our service as fake reviewer, we identified that at least

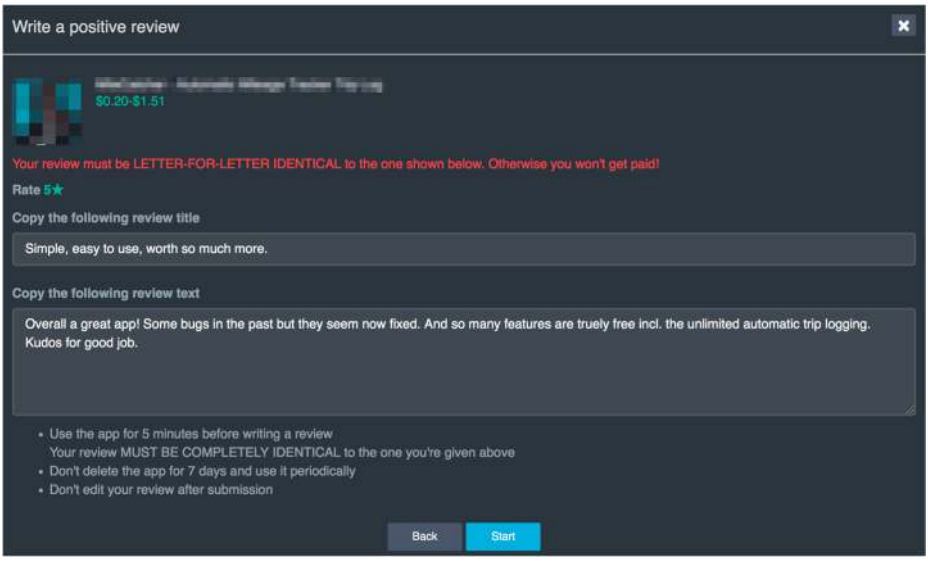

Fig. 4 A fake review request with predefined rating and review on a review exchange portal

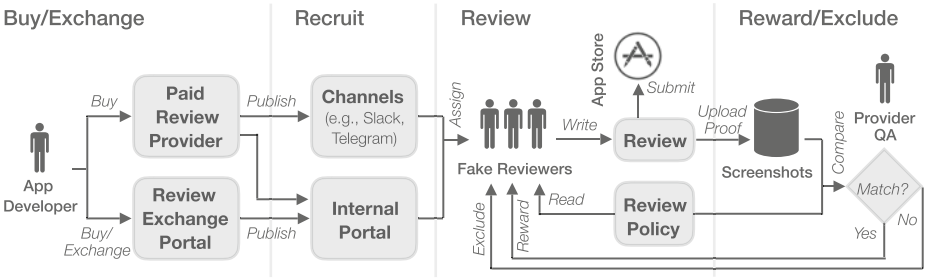

Fig. 5 Fake reviewing strategies

five providers publish their review requests on invite-only Slack or Telegram channels. By observing the communication within these channels, we found that paid review providers occasionally cross-post review requests on review exchange portals while offering micro payments.

Fake reviewers can browse through and assign review requests to themselves. Afterwards, they are presented a review policy that regulates what information or rating the review should include. The fake reviewer submits an appropriate fake review to the app store. As a proof, the fake reviewer uploads a screenshot of the review edit screen showing their rating and review to the provider.

Last, the provider compares if the provided review meets the reviewing policy. If this applies, and the review has been published within the app store, the fake reviewer is rewarded. Reviewers providing reviews that do not meet the policies are excluded from the channels or portals and are not rewarded.

3.2 Offers and Pricing Models

To increase app downloads and sales, paid review providers offer fake reviews, ratings, and installs. Table 3 shows the prices of these offers for both the Android and iOS platform. The table lists the offers' minimum and maximum price, which varies, e.g., depending on the amount of reviews bought.

Paid fake reviews are offered by 17 of 34 (50%) providers for iOS and by 32 of 34 (94.1%) for Android. Reviews always include a rating. Among all offers, reviews are the most expensive. The price of a review for iOS is, on average, between $3.41 and $4.24 with a standard deviation from 1.85 to 2.21. The price of a review for Android is less expensive, on average, between $1.73 and $2.14 with a standard deviation from 1.73 to 2.14. The price for iOS is ∼97 to 98% higher.

Paid fake ratings are offered by two of 34 (5.9%) providers for iOS and by 10 of 34 (29.4%) for Android. With this offer fake reviewers rate the app without submitting a written review. The price of a rating for iOS is on average between $1.50 and $2.00 with a standard deviation from 0.01 to 0.71. The price of a rating for Android is on average between $0.76 to $0.83 with a standard deviation of 0.76 to 0.83. Compared to Android the price for iOS ratings is ∼97 to 140% higher.

Paid installs are generated, e.g., by advertising the app on blogs. Also, users can be paid to install the app. The acquired new app users decide by themselves to rate and review the app. According to our definition these reviews are not considered as fake, as these are not

Table 3 Offers and prices (in US$) of paid review providers

PRP	Co.	Review Price				Rating Price				Install Price			
		iOS		Android		iOS		Android		iOS		Android	
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
1	IN			1.35	1.50								
2	DK	4.63	4.90	0.98	1.00					0.25	0.25	0.05	0.06
3	IN			0.25	0.25			0.20	0.20			0.09	0.09
4	IN			1.50	1.98								
5	GB			1.11	1.50							0.10	0.10
6	US			2.90	2.95			1.28	1.58			1.30	1.36
7	RU			0.25	0.25			0.20	0.20			0.10	0.10
8	US	6.00	9.00	**6.00**	**9.00**								
9	US	3.33	4.17	1.00	1.50							0.09	0.15
10	NL			1.55	1.55			1.00	1.00	0.65	0.65	0.20	0.20
11	US	2.50	4.00	3.50	5.00					0.49	0.90	0.08	0.12
12	CA			1.00	1.00								
13	US	2.15	2.50	1.59	2.50					0.34	0.46	0.13	0.20
14	US	4.30	5.00	0.85	1.20					0.35	0.38	0.35	0.38
15	IN			0.15	0.15			0.08	0.08			0.10	0.10
16	RU	2.09	2.99	2.99	2.99								
17	US	5.02	5.20	2.00	2.60					0.40	0.45	0.40	0.46
18	DE			2.50	2.50							0.17	0.17
19	US	**8.69**	**10.00**	3.60	4.00			1.28	1.60			1.36	1.58
20	VN			0.05	0.05			0.05	0.05	0.10	0.10	0.05	0.05
21	US	2.00	2.00	1.40	2.00								
22	US			1.45	2.00					0.29	0.32	0.08	0.15
23	RU	3.40	4.00	2.75	2.75								
24	US			1.00	1.00			0.80	0.80			0.15	0.15
25	NL	1.78	3.30	1.78	3.30					0.50	0.50	0.08	0.10
26	RU	3.00	3.00										
27	IN			1.99	2.40					0.39	0.46	0.39	0.46
28	CN	2.09	2.99	2.39	2.99	1.00	1.99						
29	SG	3.00	3.00	3.00	3.00								
30	DE	1.93	4.00							0.45	0.50	0.06	0.14
31	US			1.00	2.00							0.50	1.00
32	IN			0.50	0.50								
33	AE			0.90	1.00			0.75	0.80			0.15	0.40
34	IN	2.00	2.00	2.00	2.00	**2.00**	**2.00**	**2.00**	**2.00**	**1.60**	**1.67**	**1.60**	**1.67**
NUM		17	17	32	32	2	2	10	10	12	12	23	23
AVG		3.41	4.24	1.73	2.14	1.50	2.00	0.76	0.83	0.48	0.55	0.33	0.40
SD		1.85	2.21	1.24	1.70	0.71	0.01	0.64	0.71	0.38	0.40	0.45	0.50

The bold emphasis highlights the highest values per column

directly requested or paid for. Installs are offered by 12 of 34 (35.4%) providers for iOS and by 23 of 34 (67.6%) for Android. Among all offers installs are the least expensive. The price of an iOS app install is between $0.48 to $0.55 with a standard deviation of 0.38 to 0.40. For Android the price is between $0.33 to $0.40 with a standard deviation of 0.45 to 0.50. Comparing both platforms the price difference is ∼37 to 45%.

Overall, the offers are rather expensive, e.g., compared to an average crowdsourcing task or buying followers on Twitter. 10,000 Twitter followers can be bought for a price of $4 (Stringhini et al. 2013). This might depend on the fact that fake ratings and reviews are generated manually, e.g., due to a strict moderation by app store operators, while Twitter followers can be generated automatically.

We further tried to identify the popularity of the three different types of offers. Unfortunately, we were only able to extract usage numbers from paid review provider 10 (PRP10) and thus cannot provide generalizable information. Overall, this provider sold 354,000 offers, of which 20,750 (5.9%) were fake reviews, 29,150 (8.2%) fake ratings, and 304,100 (85.9%) paid installs. We cannot give any numbers on how many paid installs result into a rating or review, and if these are comparable to fake ratings and reviews.

3.3 Pretended Fake Review Characteristics

To understand the rules and conditions of providing fake reviews, we conducted a disguised questionnaire with the paid review providers. We also extracted the policies, with which submitted reviews must comply in review exchange portals.

3.3.1 Disguised Questionnaire

The disguised questionnaire consists of eleven questions and was presented to providers in a request for buying fake reviews. A sample question is shown below:

> We have several competitors which gain more and more market share. For this reason we are looking for both positive and negative reviews, positive for our apps and negative for our competitors' apps. [...]

We decided against open questions as we noticed during a pre-run of the questionnaire, conducted using different identities, that providers returned incomplete answers. The questionnaire is included in our replication package.

Eleven out of 34 paid review providers (32.3%) answered our questionnaire. Table 4 summarizes their answers. Even upon request, not all providers answered all of our questions. Therefore, the total answers refer to the number of providers that explicitly answered the specific question.

While all 11 providers offer positive ratings and reviews, 6 also offer negative, e.g., to lower the reputation of competing apps. Regarding the content, 6 of 8 providers reported to accept keywords, which will be included in their reviews. Seven of 8 providers accept predefined reviews to be submitted by their reviewers. All providers state their reviews are written by humans and not generated using algorithms. Five of 10 providers gave a guarantee to replace deleted fake reviews.

Regarding the geographical origin of fake reviews, PRP10 and PRP15 provide reviews from the US. PRP23 and PRP26 additionally provide reviews from Russia. P25 also provides reviews from India. PRP28 specified 13 countries from which the reviews are submitted, these are Austria, Canada, China, France, Germany, India, Italy, Japan, Russia, Taiwan, United Kingdom, United States, and Vietnam. Four providers (PRP9, PRP12,

Table 4 Summary of disguised questionnaire showing offers of paid review providers

PRP	Positive ratings	Negative ratings	Custom keywords	Predefined reviews	Real users	Guarantee
9	Yes	No	Yes	Yes	Yes	Yes
10	Yes	Yes			Yes	No
12	Yes	Yes	Yes	No	Yes	Yes
15	Yes	Yes				No
16	Yes	No	Yes	Yes	Yes	No
22	Yes	Yes			Yes	
23	Yes	No	No	Yes	Yes	No
25	Yes	Yes	Yes	Yes	Yes	Yes
26	Yes	Yes	Yes	Yes	Yes	Yes
28	Yes	No	No	Yes	Yes	No
29	Yes		Yes	Yes	Yes	Yes

PRP16, and PRP29) reported to submit reviews from all over the world. According to the results, the top three countries reviews are provided from are: United States (54.5%), Russia (36.4%), and India (18.2%). Regarding the language, PRP9 and PRP29 reported to provide reviews in all languages. PRP10, PRP12, PRP15, PRP25, and PRP28 only provide reviews in English. PRP23 and PRP26 also provide reviews in Russian language. By analyzing all 60,431 fake reviews, initially collected, using LangID, we found that these are written in 70 languages. The five most common are English (87.7%), French (2.3%), German (2.2%), Italian (1.3%), and Spanish (1%).

3.3.2 Review Policies

For all review exchange portals, we were able to extract policies that state the requirements submitted fake reviews have to confirm to, see Table 5. The policies have different levels of details. Thus, not every requirement is stated by each policy. With the total number we refer to the policies that explicitly state a requirement.

Table 5 Review characteristics extracted from policies of review exchange portals

REP	Co.	Real Dev.	Install App	Use App	Keep App	Honest	Rating	Length	Copy
1	IN		Yes	No			1–5		
2	ES	Yes	Yes		5 days	Yes	3–5	>10 words	No
3	US	Yes	Yes		1–2 days	Yes		2–3 sentences	
4	US							1–2 sentences	No
5	GB	Yes	Yes		1 day	Yes	4–5	1–2 sentences	
6	CN	Yes	Yes	4 min	5 days				
7	GB					Yes		1–2 sentences	
8	SE	Yes	Yes		2 days	Yes	3–5	>10 words	No
9	RU		Yes	10 min	7 days				

Five portals require to use a real device to submit a review. The installation of the app is explicitly required by seven portals. Only two portals request the reviewers to use the app before submitting a review. REP6 requires the reviewer to use the app for at least 4 min and REP9 for at least 10 min. REP1 explicitly states that the usage of the app is not required. Six portals state that the app should be kept on the phone for a specific amount of time after leaving the review. The minimum amount of time is one (REP3, REP5) up to 7 days (REP9).

Regarding the rating, four providers specify a range the rating should follow. REP2 and REP8 request 3–5 stars, and REP5 4–5 stars. REP1 is the only provider explicitly allowing positive and negative (1–5 stars) ratings. However, reviews from REP1 are not included in our fake reviews dataset. Five providers state that the review should be honest, although three of those allow only positive ratings with at least 3–4 stars. These providers state that reviewers should skip apps if they are unable to submit a positive review.

Regarding the review length, six portals make a statement: two require at least 10 words, three portals 1–2 sentences, and one portal 2–3 sentences. Three providers explicitly state that the review should not contain content copied from the app description. REP5 and REP9 additionally require that the reviews are sufficiently detailed, e.g., "should describe app features instead of providing only praise".

REP2 allows reviewers to only rate up to 10 apps per day. Further, the app should not be immediately reviewed after its installation. Reviewers should randomly download apps from the app store without leaving a review. Before leaving another review the reviewer should wait a few minutes. Last, reviewers should not only provide 5-star ratings, but vary between 3–5 star ratings. REP7 requires the ratings to match review content. Finally, REP9 requires to launch reviewed apps periodically within the next 7 days. A possible reason for this, might be to hide the suspicious behavior of quitting to use an app after providing a positive review from app store operators.

3.3.3 Initial Fake Review Indicators

Considering the questionnaire results, the review policies, and taking into account the efforts of providers to disguise fake reviews, we can hypothesize that fake reviews are highly diverse. For example, the *rating* of fake reviews can be positive or negative. The *length* of a review can also vary. In addition, the quality of the *content* may strongly differ. Overall, fake reviews do not mean short and low quality reviews, as our initial results reveal. These reviews could be either written by paid reviewers (whether or not they have to use specific keywords), or they can be predefined reviews that are written by the app developers and that have to be published by the reviewers.

4 Fake Review Characteristics (RQ2)

We investigate apps affected by fake reviews, reviewers providing fake reviews, and fake reviews themselves. In particular, we study the differences of fake reviews to the reviews from the official reviews dataset.

4.1 Apps

We identified 3,072 apps requesting fake reviews. As these apps could have, e.g., been entered on review exchange portals for testing purposes either or not by their developers, we

only consider apps that received fake reviews to strengthen our results. Overall, we analyzed 1,929 (62.8%) of the identified apps.

Most apps with fake reviews fall into in the category games, nearly twice as much as regular apps. Table 6 lists the 25 app categories of the Apple App Store. It compares apps from the fake reviews and the official reviews dataset per category. The table depicts each categories' rank, number and percentage of apps, and percentage of reviews within the datasets. The highest rank is assigned to the category with the most apps included.

We found that more than half of the apps with fake reviews (53%) belong to the category "Games", followed by the categories "Photo & Video" (5.8%), "Education" (4.8%), "Entertainment" (4.5%), and "Health & Fitness" (4.4%). The categories with least apps with fake reviews are "Stickers" (0.05%), "News" (0.1%), "Catalogs" (0.16%), "Newsstand" (0.16%), and "Books" (0.21%).

Table 6 Category ranking, number of apps, and percentage of reviews per category within the fake reviews and official reviews dataset

Category	Fake Reviews Dataset			Official Reviews Dataset		
	Rank	Apps	Reviews	Rank	Apps	Reviews
Books	21	4 (0.21%)	0.06%	19	25069 (1.75%)	0.89%
Business	12	33 (1.71%)	1.72%	3	130825 (9.15%)	1.35%
Catalogs	23	3 (0.16%)	0.09%	23	10951 (0.77%)	0.29%
Education	3	92 (4.77%)	3.80%	2	131302 (9.18%)	1.74%
Entertainment	4	87 (4.51%)	4.03%	5	79504 (5.56%)	5.68%
Finance	17	17 (0.88%)	1.11%	14	34684 (2.43%)	1.66%
Food & Drink	15	19 (0.98%)	0.65%	9	50944 (3.56%)	1.34%
Games	1	1023 (53.03%)	47.57%	1	326864 (22.86%)	49.95%
Health & Fitn.	5	85 (4.41%)	5.81%	8	54410 (3.80%)	3.23%
Lifestyle	8	69 (3.58%)	4.82%	4	102183 (7.15%)	2.46%
Medical	20	13 (0.67%)	0.65%	16	30101 (2.10%)	0.42%
Music	11	36 (1.87%)	1.99%	11	43874 (3.07%)	2.72%
Navigation	20	13 (0.67%)	1.06%	20	21559 (1.51%)	0.80%
News	24	2 (0.10%)	0.06%	18	26358 (1.84%)	1.64%
Newsstand	23	3 (0.16%)	0.18%	25	1021 (0.07%)	0.00%
Photo & Video	2	112 (5.81%)	7.66%	12	40034 (2.80%)	6.54%
Productivity	10	42 (2.18%)	1.92%	10	44191 (3.09%)	3.35%
Reference	18	14 (0.73%)	0.63%	15	34465 (2.41%)	1.50%
Shopping	13	25 (1.30%)	2.50%	22	15253 (1.07%)	2.15%
Social Netw.	6	82 (4.25%)	3.69%	17	27488 (1.92%)	5.41%
Sports	9	45 (2.33%)	1.79%	13	37060 (2.59%)	1.14%
Stickers	25	1 (0.05%)	0.01%	21	20979 (1.47%)	0.01%
Travel	14	22 (1.14%)	2.08%	7	64846 (4.53%)	1.30%
Utilities	8	69 (3.58%)	4.69%	6	71680 (5.01%)	3.61%
Weather	16	18 (0.93%)	1.43%	24	4446 (0.31%)	0.82%
		$\sum = 1,929$			$\sum = 1,430,091$	

Between both datasets, we found a strong, positive correlation for the distribution of apps over the categories. We compared the category ranks using the Spearman rank correlation coefficient ($r_s = 0.74$, two-tailed p-value = 0.00002). The coefficient is used to measure the rank correlation, i.e., the statistical dependence between the rankings of two variables.

To identify categories with the highest difference between both datasets, we calculated the rank difference using the following formula.

$$r_{\text{diff}} = |rank_{\text{official}} - rank_{\text{fake}}| \quad (1)$$

The highest differences exis for the categories "Social Networking" ($r_{diff} = 11$), "Photo & Video" ($r_{diff} = 10$), "Business" ($r_{diff} = 9$), and "Shopping" ($r_{diff} = 9$). The 3rd category "Business" within the app store is, for example, only ranked 12th in the fake reviews dataset. Vice versa, the category "Photo & Video" contains more apps in the fake reviews dataset.

Apps with fake reviews are on average three times less offered as paid, compared to regular apps. Developers might invest a lot of money buying fake reviews. Therefore, we analyzed the monetization of apps in the fake reviews dataset. We focused on the app price and in-app purchases.

Regarding the app price, we found that 1,799 apps (93.3%) are offered for free, while 130 apps (6.7%) are paid. In comparison, the app store includes 1,167,377 (81.6%) free and 262,714 (18.4%) paid apps. The mean price of an app is \$2.16 with a standard deviation of 2.5. For the app store the mean price is \$4.07 with a standard deviation of 16.7. This difference between the mean prices is statistically significant (two-sample t-test, $p < 0.001$, CI = 0.99). However, the magnitude between the differences is slightly below small, found by calculating the effect size ($d = -0.160$) (Cohen 1988). Of the paid apps, 62 apps (47.7%, cf. 39.6% in app store) are offered for \$0.99, 39 apps (30%, cf. 18.9% in app store) for \$1.99, and 16 apps (12.3%, cf. 16.2% in app store) for \$2.99. The remaining apps (10%) cost between \$3.99-\$24.99. In the app store the price of the remaining apps (25.4%) is between \$3.99-\$999.99.

In-app purchases are offered by 759 fake-reviewed apps (39.4%). These apps contain 3,845 in-app offers, of which 3,186 are in-app purchases. On average each app includes around 4.2 in-app purchases with an average price of \$10.46. 26.33% of the in-app purchases are offered for \$0.99, 26.9% are in the range of \$1.99−\$2.99, 26.6% in the range of \$3.99−\$9.99, and 20.2% in the range of \$10.99−\$399.99. We were unable to automatically crawl in-app purchases, therefore we cannot compare the figures from the fake reviews to the official reviews dataset.

Also, apps could be further monetized through advertisements. We were unable to study this aspect, since no publicly available data on the number of advertisement impressions and revenue generated per impression exists.

Most apps targeted by fake reviews have 2–9 reviews, which is the case for 42.2% of all apps. We analyzed the total number of reviews for apps affected by fake reviews and apps from the official reviews dataset. Figure 6 groups both types of apps into given ranges of fake and official reviews. The result indicates that fake reviews are not necessarily limited to a small and specific group of apps, but could be distributed across the majority of apps.

Only less than 7% of apps affected by fake reviews were removed from the Apple App Store. We studied whether apps with a high percentage of fake reviews rather get removed from the app store, compared to apps with less fake reviews. Therefore, we crawled the apps affected by fake reviews again after three months in June 2017. Of the 1,929 apps, 131 (6.8%) were no longer available on the app store. Most of the deleted apps (68%) belong to the category "Games", 5% to "Entertainment", and 5% to "Utilities". Since no

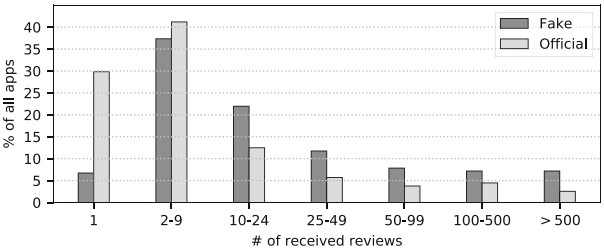

Fig. 6 Distribution of fake and official reviews over ranges per app

justification is provided by the app store operators, there are two possible reasons for this: Either, the apps have been removed by their own developers, or the app store operators have removed the app due to fake reviews or other compliance reasons, e.g., spam apps (Seneviratne et al. 2017). Figure 7 shows two plots of deleted and non-deleted apps and their percentage of fake reviews.

The upper plot considers all apps affected by fake reviews. We found that deleted apps received 27.5% fake (median: 20%) and 72.5% official reviews. Non-deleted apps received 38.1% fake (median: 30.8%) and 61.9% official reviews. By analyzing the median, we found that non-deleted apps receive 12 reviews (cf. 15 reviews for deleted apps) of which 2 are fake (cf. 2 reviews for deleted apps). A χ^2-test showed that being no longer available on the app store and the percentage of fake reviews are independent ($\chi^2 = 2.0906$, $p = 0.1482$).

As this gives the impression that the amount of fake reviews does not impact being removed from the app store, we further analyzed apps with at least ten fake reviews only, see lower plot. This applies for 181 apps, of which 11 were deleted. For these, the median of fake reviews for deleted apps is 63.5%. For non-deleted apps the median is 37.1%. Based on medians, deleted apps receive 51 reviews of which 22 are fake. Non-deleted apps receive 49.5 reviews of which 15 are fake. For these apps, a χ^2-test showed both values are no longer independent ($\chi^2 = 6.8708$, $p = 0.008762$), compared to considering all apps with at least one fake review.

4.2 Reviewers

Fake reviewers submit about 30 reviews on average — 12 times more than regular reviewers. We identified 721 users providing fake reviews. These fake reviewers provide 29.9 reviews per user, on average, compared to 2.5 reviews per reviewer in the official reviews dataset. This difference is statistically significant (two-sample t-test, $p < 0.001$, $CI = 0.99$) and the effect size is large ($d = 0.802$). Overall, these users provided 21,581

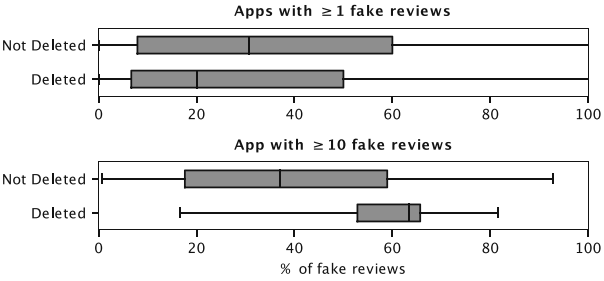

Fig. 7 Ratio of fake reviews for non-/deleted apps

reviews in total for 8,429 different apps. Surprisingly, fake reviewers do not seem to use several accounts to hide their activities.

More than 50% of the reviewers in the official dataset provide only a single review. The total number of reviews given per fake reviewer varies between 1 and 573. For reviewers within the official reviews dataset this is between 1 and 913. Figure 8 groups both fake and regular reviewers according to their number of submitted reviews.

Exactly one review was given by 5.4% of the fake reviewers, compared to 53.1% for regular reviewers. 2–5 reviews were provided by 15.8% fake reviewers (cf. 35.6%), 6–10 reviews by 20.8% (cf. 9.3%), 11–50 reviews by 40.8% (cf. 1.9%), 51–100 reviews by 11.8% (cf. 0.02%), and more than 100 reviews by 5.4% (cf. 0.006%). The highest percentage of fake reviewers (40.8%) is in the range of 11–50 reviews, while most regular reviewers (53.1%) provide a single review.

Fake reviewers review about 4 times more frequently than regular reviewers. Fake reviewers are more active compared to others. They have a frequency of one review per 78.8 days, compared to 328.9 days for regular reviewers. The difference is statistically significant with 250.1 days, i.e., 417.2% (two-sample t-test, $p < 0.001$, CI = 0.99). The effect size is large ($d = -0.955$).

The lifetime of fake reviewer accounts is nearly twice as long as regular users. The account lifetime, i.e., the time difference between the first and last review provided, is 622.3 days for fake reviewers, compared to 331.3 days for other app store users. The difference between fake and regular reviewers is 291 days (187.9%) and statistically significant using the previous test ($p < 0.001$, CI = 0.99). The effect size is near medium ($d = 0.464$). This shows that the accounts of fake reviewers remain undetected in app stores for several years.

4.3 Reviews

Although we found 60,431 fake reviews, in the following we only consider the 8,607 fake reviews that we identified and still exist in the Apple App Store. These reviews have not been filtered by mechanism of the app store operators and could impact app developers and users.

The distribution between ratings of fake and official reviews varies most for 1-star reviews. Figure 9 compares the distribution of ratings for reviews from the fake and official reviews dataset. 70% of the fake reviews are rated with 5 stars compared to 65% for official reviews. 23% of fake reviews are rated with 4 stars (cf. 16%), 5% with 3 stars (cf. 6%), 1% with 2 stars (cf. 4%), and 0.6% with 1 star (cf. 10%). Overall, ratings are very positive in both datasets. The greatest difference between fake and official reviews can be observed by the percentage of 1-star ratings. We have evidence that fake review providers explicitly ask their reviewers within reviewing policies to not only provide 5-stars reviews but also 4-stars and even 3-stars reviews. This might result in rather small differences between fake and official reviews regarding extremely positive ratings. Thereby, the suspicious behavior

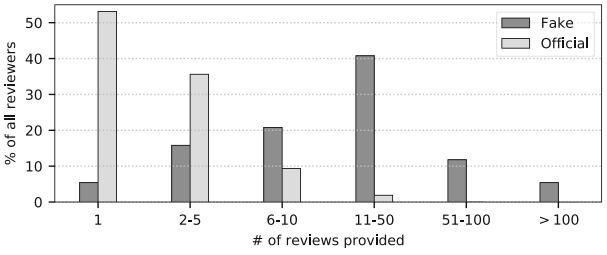

Fig. 8 Number of reviews provided per fake or official reviewer

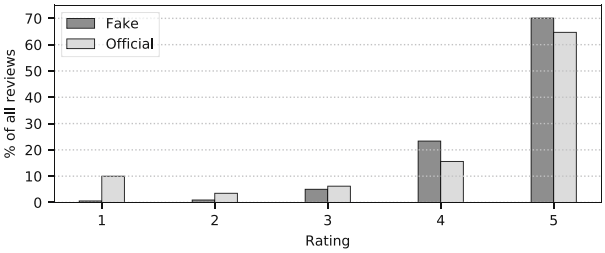

Fig. 9 Distribution of star ratings between official and fake reviews

of writing, and also receiving, only 5-stars reviews should be hidden as this could possibly result in the deletion fake reviews by app store operators and in worst case the removal of the affected app from the app store (cf. Section 3.3.2).

Compared to official reviews, short reviews are rather uncommon in fake reviews. The length of a fake review (consisting of title and body) is, on average, 121.3 characters. Official reviews have a length of 110.8 characters, on average—resulting in a difference of 10.5 characters. Considering the median, fake reviews consist of 111 characters while official reviews consist of 63 characters, see Fig. 10. The difference regarding the median is 48 characters.

We further analyzed the number of words per review. Fake reviews have, on average, 22.9 words, with a median of 21 words. Official reviews have 21.3 words, with a median of 12 words. Regarding the amount of average words, the difference is relatively small with 1.7 words. Considering the median the difference is 9 words.

A typical fake review is given below.

> Great for expense tracking ⋆⋆⋆⋆⋆
> *Does a great job for expense tracking. Nice interface and color scheme. Definitely recommend!*

We found that rather short reviews, which constitute a major part of the official reviews, are uncommon for fake reviews (see example below).

> Fantastic ⋆⋆⋆⋆⋆
> *Great game, my son loves it. Lots of fun.*

We initially assumed fake reviews to be short. However, according to the dataset fake reviews are significantly longer regarding the number of characters and words (Wilcoxon rank sum test, $p < 0.001$, $CI = 0.99$). The effect size (Fritz et al. 2012) however is near zero ($r = 0.001$).

Fake reviews are rated more often helpful compared to official reviews. In app stores users can rate the helpfulness of reviews through votes. 132 of the 8,607 fake reviews (1.5%)

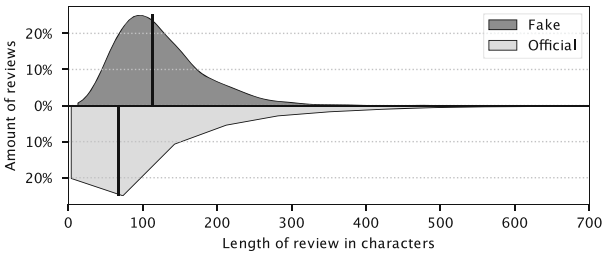

Fig. 10 Distribution of review length (in characters) between regular and fake reviews

received at least one vote, compared to 2.7% for reviews in the app store. Overall, the reviews received 270 votes. 245 are votes (90.7%) rating the reviews as helpful and 25 votes (9.3%) rate the reviews not helpful. In the app store less helpful votes (67.8%) exist. Both, the number of reviews with votes and the overall number of helpful votes are significantly different (two-sample t-test, $p < 0.001$, $CI = 0.99$). Also in this case, the effect size is near zero ($d = -0.018$).

After releasing updates, apps affected by fake reviews do not immediately receive more reviews. We analyzed the relative reviewing frequency by summing up all reviews per day after the apps' last releases. Figure 11 shows the percentage of received reviews per day for apps affected by fake reviews and regular apps over a time span of three weeks. After three weeks the amount of received reviews stabilized. We choose only the apps' last release as we were unable to automatically crawl release dates.

For regular apps most reviews are given on the day of the app release (Pagano and Maalej 2013). For apps affected by fake reviews there is only a small peak on the app release day. For these apps the percentage of reviews provided increases on a daily basis, until it decreases on day 11 after the app update. One reason might be that for apps affected by fake reviews no large user basis exist that could, intrinsically motivated, provide reviews. Developers of these apps have to buy fake reviews to promote their updates. The distribution of request for providing fake reviews to the actual reviewers might take time. Compared to that, regular apps with a user basis that matches the amount of reviews received, have enough users that spontaneously provide their feedback after installing the app update.

Fake reviews include more positives adjectives and less negative words related to software engineering such as "fix" or "crash". We analyzed the review content by comparing the 100 most-common words of fake and official reviews. We extracted the lists of most-common words in five steps. First, we removed the punctuation. We transformed all words into lowercase writing. Then, we tokenized the words of each review. We removed stopwords. Last, we counted the occurrences of each word.

Both lists have 63 words in common and 37 unique words. We sorted the lists descending by the occurrences of words. Afterwards, for each word the lists have in common, we calculated the difference between their positions in the lists. The word "simple" is, e.g., on position 14 for fake reviews and on position 97 for official reviews, resulting in a rank of −83. Therefore, negative ranks denote words that are more common for fake reviews. We plotted word ranks in Fig. 12.

The top five words that are more common for fake reviews are "simple", "super", "little", "recommend", and "well". The top five words that are less common for fake reviews are "even", "can't", "don't", "want", and "free".

Afterwards, we analyzed the distinct words in both lists. For official reviews the distinct five most common words are (in order): "update", "ever", "please", "fix", "every".

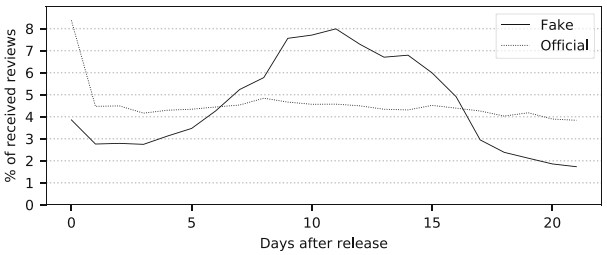

Fig. 11 Percentage of reviews received per day after app release (day 0 is release of app update)

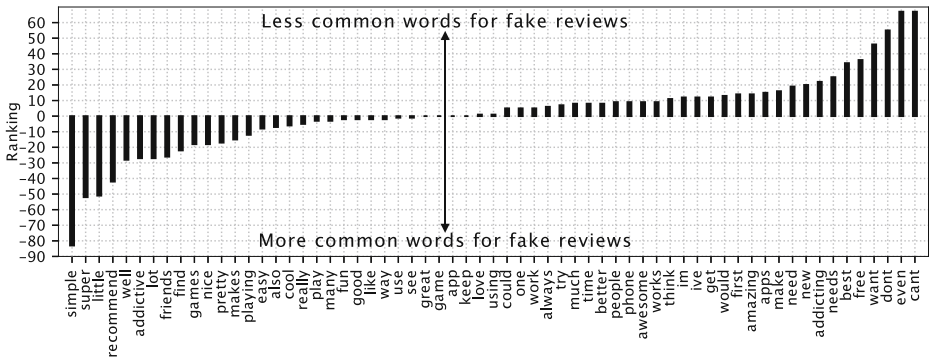

Fig. 12 Delta between occurrences of most-common words in official/fake reviews, a negative ranking indicates that the specific word is more common for fake reviews, a positive ranking denotes that the word is less common for fake reviews

Also, words possibly related to the functionality of the apps, such as "doesn't", "crashes", "wish", and "bad" are included. The five most common distinct words for fake reviews are: "graphics", "useful", "idea", "ads", and "kids". Also positive words, such as "interesting", "perfect", "helpful", "recommended", "funny", and "learn" are popular.

Last, we compared the most common bi-grams for fake and official reviews. As for most common words, again 63 matches exist. We observed that bi-grams possibly pointing to bug reports, such as "please fix", only exist in the official reviews dataset. Bi-grams indicating feature requests, such as "would like" or "wish could", exist in both datasets. Negative bi-grams, such as "waste time" or "keeps crashing", again only exist within the official reviews dataset.

5 Fake Review Detection (RQ3)

We build a supervised binary classifier to classify reviews as fake or not. Figure 13 shows the three phases conducted after feature extraction. We begin by preprocessing the data. Then, we compare the results of different classification algorithms. We optimize the algorithms by feature selection and hyperparameter tuning. Last, we evaluate the importance of the

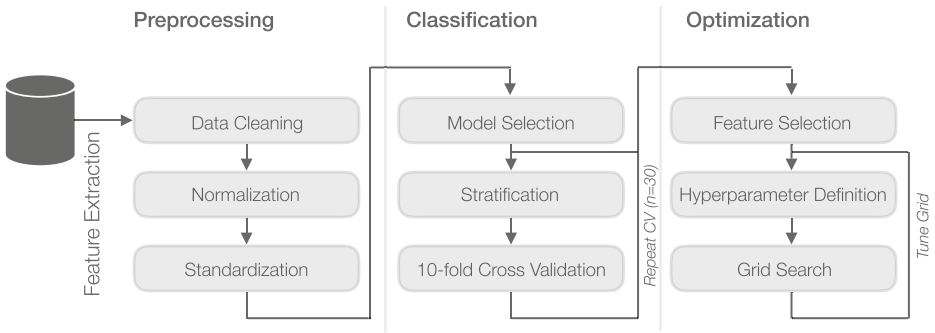

Fig. 13 Overview of fake review classification

classification features. Afterwards, we conduct an in-the-wild experiment to evaluate how our classifier performs in practice, i.e., on imbalanced data.

5.1 Feature Extraction

We extracted a balanced truthset of 16,000 reviews. Of these reviews, 8,000 are randomly selected fake and 8,000 are randomly selected official reviews. Per review, the truthset includes a vector containing 15 numerical features and a label, which is either "real" or "fake". The features were selected based on the differences we identified between fake and official reviews, and by experimentation. We decided **not** to use the textual review itself, e.g., in form of TF/IDF representation, for several reasons. Mukherjee et al. (2013b) analyzed fake reviews published on Yelp and found that the word distribution of fake reviews does not significantly differ from official reviews. As a result, their text-based classifier only achieved an accuracy of 67.8%. Vice versa, the word distribution within fake reviews could also highly differ. According to the questionnaire with paid review providers, custom keywords can be included in the reviews or predefined reviews can be submitted by the fake reviewers. Finally, when using text, training data would be required for every language, which is difficult to collect.

In contrast, Ferrara et al. (2014) use non-textual features related to the user, such as the account creation time or total number of followers. By using such features their classifier to detect bots in social networks that, e.g., influence political discussions, achieved better results. Other researchers (Dickerson et al. 2014; Lee et al. 2010; Feng et al. 2012; Park et al. 2016) followed this approach and were also able to improve their classification results.

We therefore focused on features that relate to the context of the fake review, i.e., the reviewer and app. Table 7 lists all selected features. For the reviewer we selected four features: the total number of reviews provided, the percentage of reviews per star rating (e.g., the reviewer could have provided 70% of all reviews with a 5-star rating and 30% with a 1-star rating), the review frequency (i.e., the average time in seconds between all reviews provided), and the account usage (which is the lifetime of the reviewers account, i.e., the timespan between the first and the last review provided in seconds). For the app we selected two features: the total number of reviews received for all app versions and the percentage of reviews received per star rating. Finally, as features for the review we selected the length, i.e., the characters count.

Table 7 Features selected for the classification of fake reviews

Category	Name	Type	Null-Values	Example
Reviewer	# Reviews (Total)	Int	0	100
	% Reviews (per Star-Rating)	[Float]	0	[0.7, 0.0, 0.0, 0.0, 0.3]
	Review Frequency (in Seconds)	Int	1,734	100
	Account Usage (in Seconds)	Int	0	600
App	# Reviews (Total)	Int	0	100
	# Reviews (per Star-Rating)	[Float]	0	[0.2, 0.2, 0.2, 0.2, 0.2]
Review	Length (in Characters)	Int	0	100

5.2 Data Preprocessing

We preprocessed the data in three steps. We began by performing data cleaning, i.e., filling null values instead of removing affected columns. Of the selected features only a single column includes null values, see Table 7. The review frequency is in 1,734 cases undefined because only a single review was provided by the reviewer. In this case, we set the frequency to lifetime of the app store, which is 9 years.

Then, we normalized the dataset so that individual samples to have unit norm. We used the `normalize()` method with standard parameters of the `preprocessing` module provided by scikit-learn (Pedregosa et al. 2011).

Last, we standardized the dataset so that the individual features are standard normally distributed, i.e., gaussian with zero mean and unit variance. This is a common requirement for many classification algorithms, such as the radial basis function (RBF) kernel of support vector machines. If not standardizing the data, features with a much higher variance compared to others might dominate the objective function. As a result, the classification algorithm is unable to learn from other features (SciKit 2018). We used the `scale()` method with standard parameters of the `preprocessing` module.

5.3 Classification with Balanced Data

We compare seven supervised machine learning approaches to classify reviews as fake or not. We use the implementations provided by the scikit-learn (Pedregosa et al. 2011) library. Supervised approaches need to be trained using a labeled truthset before they can be applied. This truthset is split into a training and testing set. The training set is used by the algorithms to build a model on which unseen instances are classified. In the test phase, the classifier performs a binary classification and decides whether reviews within the test set are fake or not.

To get more reliable measures of the model quality, we apply cross validation on our truthset. This is performed in several folds, i.e., splits of the data, called k-fold cross validation. In this paper we perform 10 folds. Per fold, a randomly selected amount of 1/k of the overall data is held out of the training as a test set for the evaluation. The final performance is the average of the scores computed in all folds. Using cross validation, we also avoid bias which would otherwise be introduced by using only a random train/test split. In addition, although our truthset is balanced, we apply stratification. Stratification ensures that each split contains a balanced amount of fake and official reviews. We repeat the cross validation 30 times per classification algorithm with different seeds. We use the `RepeatedStratifiedKFold` method of the `model_selection` module.

The seven classification algorithms we compare are the following: Naive Bayes (GaussianNB) is a popular algorithm for binary classification (Bird et al. 2009), which is based on the Bayes theorem with strong independence assumptions between features. Compared to other classifiers it does not require a large training set. Random Forest (RF) (Ho 1995) is an ensemble learning method for classification and other tasks. It can build multiple trees in randomly selected subspaces of the feature space. Decision Tree (DT) (Torgo 2010) assumes that all features have finite discrete domains and that there is a single target feature representing the classification (i.e., the tree leaves). Support Vector Machine (SVM) (Cortes and Vapnik 1995) represents the training data as points in space. It creates support vectors for gaps between classes in the space. The test data is classified based on which side of the gap its instances fall. The Gaussian radial basis function (rbf) is used for non-linear classification by applying the kernel trick (Aizerman et al. 1964). Linear support vector classification

Table 8 Classifiers' scores to detect fake reviews

Classifier	Accuracy	Precision	Recall	F1	AUC/ROC
RandomForestClassifier	0.970	0.973	0.967	0.970	0.989
DecisionTreeClassifier	0.953	0.949	0.957	0.953	0.953
MLPClassifier	0.919	0.921	0.916	0.918	0.969
SVC(kernel='rbf')	0.901	0.879	0.930	0.904	0.959
SVC(kernel='linear')	0.899	0.878	0.926	0.902	0.960
LinearSVC	0.895	0.861	0.941	0.900	0.964
GaussianNB	0.765	0.731	0.889	0.755	0.955

(LinearSVC) penalizes the intercept, in comparison to SVM. Multilayer perceptron (MLP) is an artificial neural network which consists of at least three layers of nodes. MLP utilizes the supervised learning technique backpropagation for training.

Table 8 shows the results of the seven classification algorithms, each with default configuration. The results include accuracy, precision, recall, F1-score, and area under the ROC curve (AUC) value. Among all, the random forest algorithm achieved the best scores.

5.4 Optimization

We optimize the classifiers by performing feature selection and hyperparameter tuning. We optimize for **precision only**. In comparison to fake reviews, for regular reviews we were unable to create a gold-standard dataset. To create a gold-standard dataset for regular non-fake reviews, all fake reviews must be identified and removed from the official reviews dataset. This is practically in-feasible as there is currently no measure to ensure that a review is not fake.

Hence we do not know all fake reviews, we can only report on how many of the *known* fake reviews are classified as fake (precision) and not on how many of *all existing* fake reviews were classified as fake (recall). Resultant measures, such as the F1-score, are reported for completeness.

To select features, we apply recursive feature elimination with cross validation. After the classification algorithm assigned a weight to each feature, these are eliminated recursively by considering smaller sets. Per iteration, the least important feature is removed to determine their optimal number. We use the `RFECV` method from the `feature_selection` module. The cross validation is performed as described in the previous phase.

We received the best result with the random forest algorithm using all features. Nearly similar accuracies are already possible with less features, e.g., the precision with three features is 0.969, compared to 0.973 using all features, see Fig. 14. The three selected features are the *1) total number of reviews the app received and 2) the user provided*, as well as the *3) frequency in which the user provides reviews*.

To tune the hyperparameters, we apply the grid search method `GridSearchCV` from the `model_selection` module. This method performs a cross validated, exhaustive search over a predefined grid of parameters for a classification algorithm. After finding the optimal combination of parameters within the grid, this is further manually tuned by adding more values around the currently best.

We achieved the best result using the random forest algorithm with the parameters {*'criterion': 'gini', 'max_depth': 30, 'max_features': 'sqrt', 'n_estimators': 300*}. The

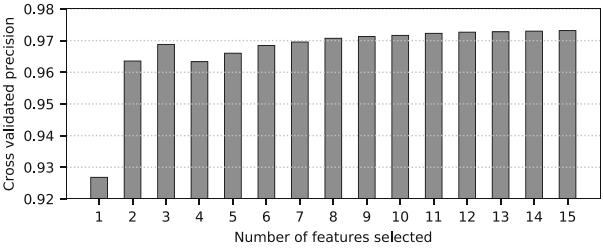

Fig. 14 Precision of random forest classifier by number of features selected

parameter *criterion* measures the quality of a split. We used Gini impurity, which is intended for continuous attributes and faster to compute compared to Entropy. It is recommended to minimize the number of misclassifications. *max_depth* defines the depth of the tree. *max_features* sets the number of features to consider when looking for the best split. *n_estimators* defines the number of trees in the forest.

Although performing hyperparameter tuning, the classifier's precision equal the results using the default configuration. Only measures we do not consider, the recall and F1-score, were slightly improved resulting in 98% each.

5.5 Feature Importance

Last, we analyzed the relative importance of the extracted features with respect to the predictability of whether a given review is fake or not, called feature importance. The feature importance is calculated on how often a features is used in the split points of a tree. More frequently used features are more important.

Figure 15 shows that the three most important feature are the total number of reviews an app received (30%), the total number of reviews a user provided (24%), and the frequency with that a user provides reviews (13%).

We assume the total number of reviews received by an app is the most important feature as apps with a specific amount of reviews, e.g., 2–9 reviews (cf. Section 4.1), are most often targeted by fake reviews. The total number of reviews a user provided as well as review frequency have a high importance as fake reviewers provide much more reviews than regular reviewers, with a higher frequency. The percentage of 1- and 2-star ratings an app received are important, as the difference between those star ratings provided are the highest when comparing apps with fake reviews and regular apps (cf. Fig. 9).

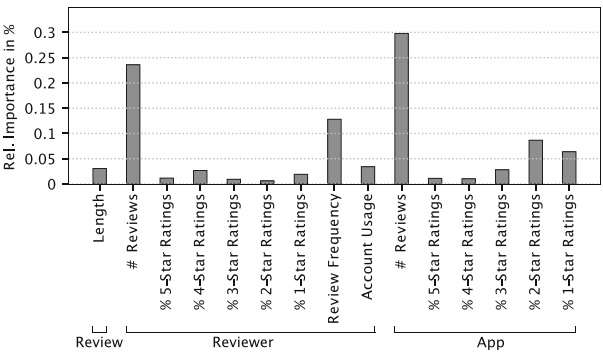

Fig. 15 Relative importance of extracted features to detect fake reviews

5.6 Classification with Imbalanced Data

In practice, fake and regular reviews are imbalanced. For app stores no reliable estimate on the distribution exists. Other domains, such as social media, mark 10% to 15% of their reviews as fake (Sussin and Thompson 2018). The travel portal Yelp filters about 15% of their reviews as suspicious (Luca and Zervas 2016; Mukherjee et al. 2013a). This class imbalance can additionally be affected by numerous factors, such as the selected apps or time period. Free apps, for example, receive more fake reviews than paid apps (cf. Section 4.1). This reveals a skewed distribution of fake and regular reviews in app stores.

Research found that highly imbalanced data often results into poor performing classification models (Drummond et al. 2003; Chawla et al. 2004; Saito et al. 2007). To have a more realistic setting of how our classifier can perform in practice, we conduct an in-the-wild experiment by varying the skewness of our dataset. We decided to vary the skewness on a logarithmic scale to depict the classification scores on finer granularities towards extremely imbalanced datasets with fake reviews as the minority class. We keep a fixed amount of 8,000 fake reviews and create 27 datasets including $10^2 - 10$ (90%) to 10^{-1} (0.1%) fake reviews. For a skew of 90% we used 889 regular reviews. With every change of the skewness we added additional regular reviews. All of the about 8 million used regular reviews were randomly selected at once from the official reviews dataset, so that the classification results are comparable.

Figure 16 shows an overview of the results of the seven supervised machine learning approaches studied in this work. Per classification algorithm and skewed dataset, we report on the precision, recall, F1-score, and AUC value. Since identifying fake reviews is a

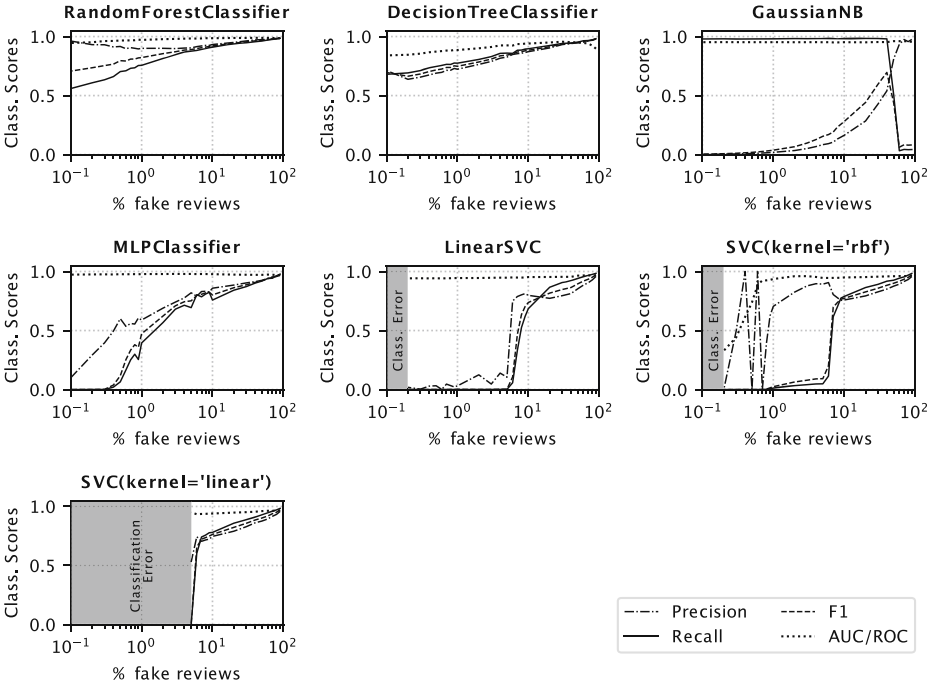

Fig. 16 Classification scores of machine learning algorithms on imbalanced datasets, including 90% to 0.1% fake reviews, plotted on a logarithmic scale

two-class problem, the performance metrics can be derived from the confusion matrix that is generated with every classification, see Table 9. We explain the reported performance metrics in the following: Precision ($\frac{TP}{TP+FP}$) measures the exactness, i.e., the number of correctly classified fake reviews to the overall number of reviews classified as fake. Recall ($\frac{TP}{TP+FN}$) measures the completeness, i.e., the number of correctly classified fake reviews to the overall number of fake reviews. The F1-score ($\frac{recall*precision}{precision+recall}$) is the harmonic mean between precision and recall. As improving precision and recall can be conflicting, it shows the trade-off between both. The AUC value measures the area under the ROC curve. It varies within the interval [0, 1]. The ROC curve itself depicts all possible trade-offs between TP rate, i.e., recall, and FP rate ($\frac{FP}{TN+FP}$). A better classifier produces an ROC curve closer to the top-left corner and therefore a higher AUC value (Wang et al. 2018).

As we are focusing on skewed datasets within our in-the-wild experiment, we need to select performance metrics that are insensitive to class imbalance. This applies for all measures that use values from only one row of the confusion matrix (Wang et al. 2018). Precision and F1-score are sensitive to class imbalance and biased towards the majority class. Therefore, these metrics are inappropriate for our evaluation. Recall and AUC/ROC value are insensitive to class distribution, we use both measures to compare the performance of the classification algorithms within our in-the-wild experiment.

In the further evaluation, we include all classifiers that achieve a recall and AUC value higher than 0.5 for datasets including 90% to 1% fake reviews. This applies for the random forest (RF), decision tree (DT), and MLP algorithm. The remaining algorithms are excluded from the evaluation. The recall of the gaussian naive bayes (GaussianNB) algorithm decreases to nearly 0 for datasets with fake reviews as the majority class. Also, its precision is extremely low for datasets including less than 10% fake reviews. Similarly, the recall of the SVC algorithms decrease to nearly 0 for datasets with less than 10% fake reviews. In addition, the SVC(kernel = 'linear') implementation of sklearn return errors for skews below 5%, so do the remaining two SVC algorithms for skews of 0.1%.

5.6.1 Classification Results with Imbalanced Data

Figure 17 shows the performance measures of all three classification algorithms that remain within our evaluation. We chose to depict each measure as a single plot to more easily compare the algorithms. We report the precision and F1-score for reasons of completeness, although these are inappropriate measures for imbalanced data. Within the graphs, we highlight in gray the interval in which fake reviews typically occur in other domains. Further, we mark where the classes are equally distributed, i.e., there exist 50% fake reviews. At this point the algorithms perform well with all measures above 0.9 (cf. Table 10).

Unfortunately, in practice there can exist imbalances towards fake or regular reviews being the majority class. From our results and research in other domains it is more likely that the bias is towards regular reviews. For this reason we choose more detailed results towards fake reviews being the minority class.

Table 9 Confusion matrix of a two-class problem

	Predicted as fake review	Predicted as regular review
Actual fake review	True positive (TP)	False negative (FN)
Actual regular review	False positive (FP)	True negative (TN)

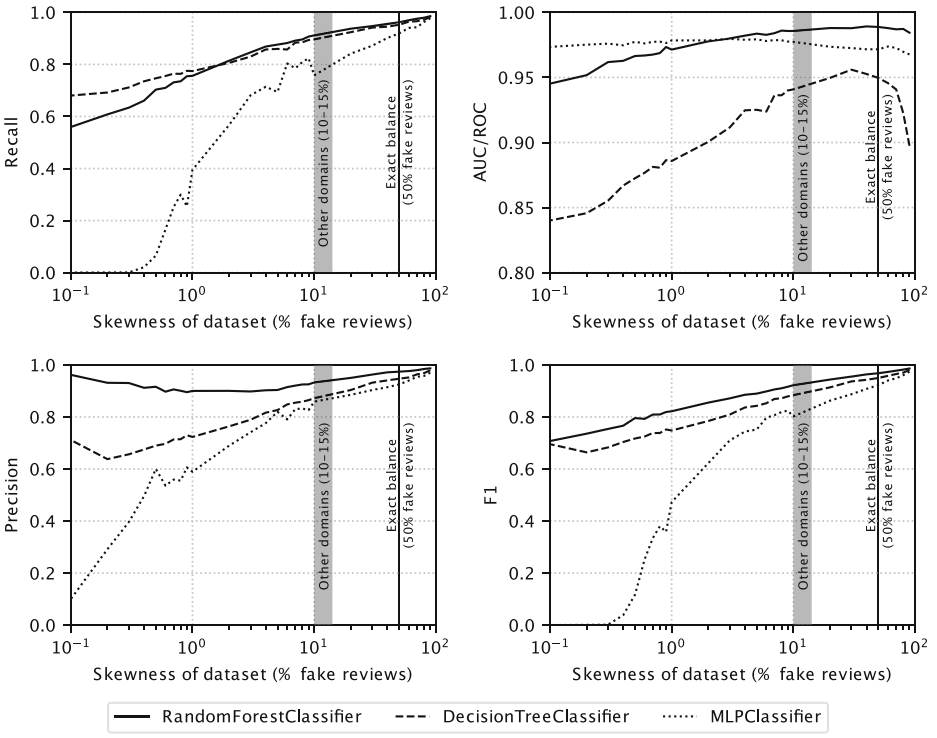

Fig. 17 Classification scores of appropriate machine learning algorithms for datasets with class imbalance, i.e., including 90% to 0.1% fake reviews, plotted on a logarithmic scale

However, when fake reviews become the majority class (towards the right of the 50% mark) the recall improves for all algorithms, by up to 6.4% for the MLP algorithm. The best result is achieved by the RF algorithm with 0.986 recall. The AUC value decreases in all cases, for the RF and MLP algorithms the value slightly decreases by up to 0.5%. The value of the DT algorithm decreases more stronly by 5.5%. The best AUC value is achieved by the random forest algorithm with 0.984.

When fake reviews become the minority class most performance measures decrease. With an amount of **10% fake reviews** (10^1), as reported to be typical for other domains (Sussin and Thompson 2018; Luca and Zervas 2016; Mukherjee et al. 2013a), the recall of the RF and DT algorithm are nearly identical (RF: 0.912, DT: 0.896). Compared to the result of the balanced dataset, the recall decreased by 5.3% for the RF algorithm and by 5.9% for the DT algorithm. The recall of the MLP algorithm is significantly less with 0.758 (−17.6%). The AUC value is the highest for the RF algorithm (0.986, +0.3%), followed by the MLP (0.977, +0.6%) and DLT (0.941, −0.9%) algorithms.

For **1% fake reviews** (10^0), the recall of the RF and DT algorithms is still nearly identical (RF: 0.756, −21.4%; DT: 0.775, −18.7%), followed by the MLP algorithm (0.395, −57.1%). The AUC value is the highest for the MLP algorithm (0.978, +0.7%), followed by the RF (0.971, −1.8%) and DT (0.886, −6.7%) algorithms.

For an amount of **0.1% fake reviews** (10^{-1}) the recall is the highest for the DT algorithm (0.680, −28.6%), followed by the RF algorithm (0.560, −41.8%). The recall of the MLP

Table 10 Classification scores on imbalanced datasets with skews of 90% to 0.1% fake reviews (DT: DecisionTreeClassifier, MLP: MLPClassifier, RF: RandomForestClassifier)

Skew	Recall			AUC/ROC			Precision			F1-score		
	DT	MLP	RF	DT	MLP	RF	DT	MLP	RF	DT	MLP	RF
90.0	0.982	0.978	0.986	0.897	0.968	0.984	0.979	0.970	0.987	0.981	0.974	0.986
80.0	0.972	0.963	0.980	0.923	0.970	0.987	0.968	0.958	0.984	0.970	0.960	0.982
70.0	0.966	0.943	0.976	0.941	0.973	0.987	0.964	0.954	0.980	0.965	0.949	0.978
60.0	0.964	0.940	0.970	0.946	0.974	0.988	0.952	0.940	0.976	0.958	0.940	0.973
50.0	0.953	0.920	0.962	0.950	0.972	0.989	0.947	0.925	0.974	0.950	0.922	0.968
40.0	0.945	0.902	0.956	0.953	0.972	0.989	0.941	0.914	0.972	0.943	0.908	0.964
30.0	0.942	0.872	0.947	0.956	0.973	0.988	0.932	0.903	0.963	0.937	0.887	0.955
20.0	0.924	0.842	0.937	0.950	0.974	0.988	0.903	0.885	0.950	0.914	0.863	0.944
10.0	0.896	0.758	0.912	0.941	0.977	0.986	0.872	0.859	0.933	0.884	0.802	0.922
9.0	0.894	0.824	0.907	0.940	0.978	0.986	0.865	0.826	0.925	0.879	0.824	0.916
8.0	0.885	0.809	0.896	0.936	0.979	0.986	0.859	0.832	0.925	0.872	0.819	0.910
7.0	0.884	0.785	0.892	0.936	0.979	0.984	0.855	0.830	0.920	0.869	0.807	0.906
6.0	0.857	0.804	0.882	0.924	0.978	0.983	0.848	0.790	0.915	0.853	0.797	0.898
5.0	0.860	0.694	0.876	0.925	0.979	0.984	0.826	0.820	0.904	0.843	0.752	0.890
4.0	0.857	0.714	0.869	0.925	0.979	0.982	0.816	0.776	0.902	0.836	0.744	0.885
3.0	0.830	0.681	0.846	0.912	0.980	0.980	0.789	0.741	0.898	0.809	0.710	0.871
2.0	0.806	0.567	0.814	0.901	0.978	0.978	0.764	0.690	0.900	0.784	0.622	0.855
1.0	0.775	0.395	0.756	0.886	0.978	0.971	0.723	0.589	0.900	0.748	0.473	0.822
0.9	0.776	0.255	0.755	0.886	0.976	0.973	0.730	0.606	0.895	0.752	0.359	0.819
0.8	0.764	0.300	0.735	0.881	0.978	0.969	0.714	0.555	0.901	0.738	0.380	0.810
0.7	0.765	0.252	0.731	0.881	0.977	0.968	0.714	0.559	0.906	0.738	0.334	0.809
0.6	0.756	0.166	0.710	0.877	0.976	0.967	0.697	0.536	0.897	0.725	0.253	0.793
0.5	0.747	0.065	0.703	0.873	0.977	0.966	0.690	0.600	0.916	0.717	0.117	0.796
0.4	0.735	0.021	0.661	0.867	0.975	0.963	0.675	0.496	0.912	0.704	0.038	0.766
0.3	0.712	0.001	0.634	0.855	0.976	0.962	0.656	0.396	0.930	0.683	0.002	0.754
0.2	0.692	0.001	0.608	0.846	0.975	0.952	0.638	0.292	0.931	0.664	0.001	0.736
0.1	0.680	0.000	0.560	0.840	0.973	0.945	0.712	0.100	0.962	0.696	0.000	0.707

algorithm dropped to 0 at about 0.3% fake reviews and below within the dataset. The AUC value is the highest for the MLP algorithm (0.973, +0.2%), followed by the RF algorithm (0.945, −4.4%). Last, the AUC value of the DT algorithm significantly decreased to 0.840 (−11.4%).

Comparing all three algorithms using their recall and AUC/ROC value, **the random forest algorithm performs best for imbalanced datasets**. Although, the decision tree algorithm achieves a better recall when the dataset is extremely skewed (less than 1% fake reviews), its AUC/ROC value is significantly lower for all datasets. Similar, the MLP algorithm achieves better AUC/ROC values for datasets with less than 1% fake reviews. However, the recall of the MLP algorithm drops to 0 for extremely skewed datasets. With skews common for other domains, the random forest algorithm performs best with a recall of 0.912 and AUC/ROC value of 0.986.

6 Discussion

We discuss implications of fake reviews on software engineering, and from the perspective of app users and store operators. Then, we discuss the results' validity.

6.1 Implications

In modern app stores, developers are for the first time able to publicly retrieve customers' and users' opinions about their software and to compare its popularity in form of rank or number of downloads. Although app reviews provide a rich source of information, they may not be fully reliable, as customers may leave reviews that do not reflect their true impressions (Finkelstein et al. 2017). Our work shed light on one of these cases: fake reviews.

Generally, fake reviews, i.e., paid, incentivized reviews (which can provided either directly or via fake review providers) are prohibited by official app store reviewing policies. The main reason is to **preserve the integrity of app stores** (Apple 2017; Google 2018). Users that do not trust app stores and their reviews will most likely refrain from providing app reviews themselves. This would harm one of the most important advantages of app stores: collecting *real, spontaneous feedback* on software in a channel used by both developers and users.

We applied our fake review classifier to the full official Apple App Store dataset. As a results, 22,207,782 (35,5%) of all 62,617,037 reviews were classified as fake. This number seems very high at first and can only be used as a first indication. Further studies need to be carried out to give a precise approximation of the amount of fake reviews in official app stores. Still, multiple indices indicates a non-trivial amount of fake reviews in app stores. We identified about 60,000 reviews from only a single provider. Overall, we identified 43 providers while much more might exists or have existed before. If every provider would provide the same number of reviews, the amount would sum up to 2.58 million fake reviews. We hypothesize that the majority of fake reviews is written by persons, who get directly asked by developers. Although not generalizable, we repeatedly observed this phenomena in our professional app development settings. When apps are developed privately, friends were asked to provide fake reviews. When programmed in a commercial environment, either employees of the developing company or of the ordering company (cf. Bell (9to5Mac 2017)) are asked to provide fake reviews. Given that 1.4 million apps exists within the dataset the number of fake reviews does no longer seem unattainably high. Such amount of fake reviews are also presumed in other domains. Streitfeld (Times 2017) report that every fifth review submitted to Yelp is detected as dubious by internal filters.

App users might, by using positive or negative fake reviews, get mislead to either downloading an app or not. As shown by Ott et al. (2011) fake reviews sound authentic and are hard to detect by humans. In an experiment, humans at most scored an accuracy of 61% identifying fake reviews, even as the word distribution of the used fake reviews differed from regular reviews. We think that this also applies for apps. Users and developers might not be able to identify fake reviews only based on their text.

Measures and tools should enable users (and developers) to identify fake reviews and affected apps. Such tools already exist for products sold on Amazon, e.g., Fakespot (2017). Users enter the name of a product to determine if its reviews are trustworthy. Fakespot also takes a step towards analyzing fake reviews in the app store. The features used to classify

reviews as fake also related to the review context, such as if a large number of positive reviews is provided within a short period of time. However, the selected criteria to classify reviews as fake are not completely transparent nor empirically validated. Also, we assume that no gold-standard dataset has been used for fake reviews. For the Instagram app the site, e.g., classifies 50% (about 600,000) of the reviews as fake, which raises accuracy concerns for this approach.

The **research area App Store Analysis** covers work to mine apps and their reviews and extract relevant information for software and requirements practitioners, e.g., to get inspirations about what should be developed and to guide the development process (Harman et al. 2012; Pagano and Brügge 2013; Pagano and Maalej 2013; Villarroel et al. 2016; Chen et al. 2014; Guzman and Maalej 2014; Iacob and Harrison 2013; Khalid 2013; Carreno and Winbladh 2013). Martin et al. provide a comprehensive literature review of this area (Martin et al. 2016). The majority of papers identified in their study (127 of 187 papers, 68%) analyze non-technical information, such as app reviews.

Review Analysis itself is one of the largest sub-fields of app store analysis, which receives a significant and increasing number of publications each year. Work in this sub-field started by analyzing the content of app reviews (2012–2013), afterwards focusing on adding additional features such as sentiments (2013–2014). Then, app reviews were summarized to extract app requirements. Although, information extracted from app reviews is getting increasingly integrated into the requirements engineering processes (Palomba et al. 2015; Maalej et al. 2016b), **none of the papers discusses the impact of fake reviews**.

In the following, we discuss the potential impact of fake reviews on software engineering along with the main review analysis topics according to Martin et al. (2016).

Fake reviews, similar to official reviews, include **requirements-related information, such as feature requests.** Oh et al. automatically categorize app reviews into bug reports and non-/functional requests to produce a digest for developers including the most informative reviews (Oh et al. 2013). Additional work focuses on extracting requirements-related information from app reviews (Iacob et al. 2013, 2014; Panichella et al. 2015). Iacob and Harrison found that 23.3% of the app reviews include feature requests (Iacob and Harrison 2013). We applied the classifier of Maalej et al. (2016a) to extract bug reports and feature request from the 8,000 official and 8,000 fake reviews included in our truthset (see Fig. 18). While fake and regular reviews are imbalanced within the overall app store dataset, when applying review analysis approaches on a subset of reviews the distribution is unknown. For this reason, we did not set a specific distribution of fake and regular reviews. Within the official reviews, we identified 1,297 bug reports and 921 feature requests, while we found that the fake reviews contain 362 bug reports and 521 feature requests. We include an example feature request within the fake review dataset below.

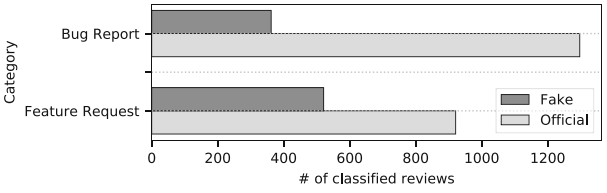

Fig. 18 Reviews within truthset classified as bug report and/or feature request

Nice UI ⋆ ⋆ ⋆⋆

Very clean and beautiful UI. I like the goal setting and the reminders. I would like to see some animation when scrolling the weekly progress bars.

We assume that most fake reviewers did not use the reviewed app before and are unfamiliar with it. In addition, review policies ask fake reviewers to explicitly talk about features rather than providing praise only. For this reasons, it is unclear if those feature requests are really relevant or only thought up to make the review sound more authentic. Fake reviews might thus impact the results of existing classifiers. When not removing or at least flagging fake reviews with information relevant for developers, wrong assumptions for the future decisions might be drawn.

Researchers showed that nearly **half of the feature requests included in app reviews are implemented.** Hoon et al. highlight that user expectations are changing rapidly, as observable through app reviews. Developers must keep up with the demand to stay competitive (Hoon et al. 2013). Palomba et al. studied the reviews of 100 open-source apps. By linking reviews to code changes the authors showed that 49% of the changes requested were implemented in app updates (Palomba et al. 2015). These results show that a significant amount of changes proposed by users are integrated into software. Some of these changes might be inspired and prioritized based on fake reviews.

Recent approaches summarize and extract requirements-related information from **app reviews of related or competing apps.** Fu et al. present WisCom to analyze app reviews per app/market level, e.g., to get an overview of competing apps (Fu et al. 2013). Gao et al. present AR-Tracker to summarize app reviews to real issues and prioritize them by their frequency and importance (Gao et al. 2015). Nayebi et al. mine app reviews and tweets of similar apps within a specific domain (Nayebi and Ruhe 2017; Nayebi et al. 2017). These approaches monitor and extract information from app reviews of related or competing apps. While developers will probably know when their apps receive fake reviews, i.e., when they bought those instead of being affected by negative fake reviews bought by competitors, developers cannot be fully sure if competing apps receive fake reviews. Wrong conclusions can also be drawn from fake reviews including the honest opinion of reviewers. Fake reviewers can just copy and modify a regular review they honestly agree with and resubmit those on review exchange portals. This way the frequency (and hence the priority) of, e.g., a feature request, might be fake (i.e. incentivized) and thus biased. Table 5 highlights that only three review exchange portals forbid fake reviewers to copy and modify existing reviews. Using those reviews as input for summarization approaches, "wrong" features could emerge as a result.

App store operators try to prevent fake reviews by providing review policies. The Apple App Store policy (Apple 2017) states that apps will be removed and that the developers may be expelled from the app store's developer program "*if we find that you have attempted to manipulate reviews, inflate your chart rankings with paid, incentivized, filtered, or fake feedback*". This is the case when app developers buy fake reviews. However, the concrete actions taken to identify fake reviews are non-transparent. We also noticed that larger, popular apps try to prevent negative feedback from being submitted to the app store. These apps ask users for submitting feedback within the app in form of star ratings. The rating is not directly forwarded to the app store. In case of a one star rating, a mail form appears asking the user to submit the review directly to the app developer instead of forwarding it to the app store, where the review is publicly visible. Such actions might also manipulate the app ratings and reviews as well.

We think that researchers should carefully sample apps and perform data cleanings before studying and mining app reviews. For example, apps with an unusual distribution of ratings might be affected by fake reviews. Similar, reviews of users with an amount or frequency above average might have to be removed or considered separately during data cleaning. Otherwise, wrong assumptions for the future development of an app could be drawn.

Collecting and analyzing **context and usage information** can help substantiate decisions and check the quality of reviews (Maalej et al. 2016b). App developers can, e.g., utilize the number of users or the average time users spend with a specific feature to decide which parts of their apps to improve and which suggestions should be taken into consideration in the next release. App store operators can take measures, such as the number of times a user opened an app or the daily app usage time to decide the trustworthiness and weight of reviews within an app's overall rating. Instead of limiting the amount of users who can participate in the reviewing process, one might think about to weight or consider incentivized reviews differently and in a transparent manner. Since the overall app store ecosystem is designed that more positive reviews lead to more downloads and thus increase the app's success (Harman et al. 2012), developers will likely continue to ask "friends" to rate their apps. Even if incentivized and not independent, such reviews can also include useful information. Instead of excluding the reviews and their reviewers, a possible alternative might be to highlight these review with badges (e.g., friend, expert, or crowd tester).

6.2 Results Validity

The results of our study might have limitations and should be considered within the study context.

The process of extracting actual fake reviews was challenging. However, we did not try to generate fake reviews using a crowdsourced approach, as we wanted to only rely on fake reviews that have been published to and still remain unidentified within the app store by its users and operators. For this reason, we decided only using a subset of our about 60,000 collected fake reviews.

Nearly all reviews were extracted from a single provider. This provider is a review exchange portal (see Table 1, REP3). On these portals reviewers could submit their honest opinion. However, **we consider the collected reviews as fake for the following reasons:**

First, app store operators strongly require that app reviews must be 1) written by *real users* of the app and 2) cannot be *incentivized*. Both conditions are not given in review exchange portals. For this reason, these reviews are fake according to the definition and agreement or app store providers. Even if reviewers are allowed to submit their honest opinion according to the review policy of this exchange portal, *rewarded, incentivized, or non-spontaneous reviews* are prohibited by the official Google and Apple App Store Review Guidelines (Apple 2017; Google 2018).

Second, review exchange portals provide predefined ratings and review messages. These ratings do not necessarily correspond to the opinion of the reviewers. The providers' review policies ensure that reviewers that post their honest opinion are not being rewarded and are excluded from reviewing portals. The review policy of REP3 does not include a general rating, e.g., 3-stars or above (cf. Table 5). The provider uses individual policies per app (cf. Fig. 4). We could not extract historical data to say if all individual policies included a predefined rating. However, for active review requests at the time of data collection, predefined ratings were included in all cases.

Third, paid review providers and review exchange portals share reviewers. As identified, paid review providers cross-post their review requests on review exchange portals (cf.

Fig. 5). This also applies for REP3. Paid review providers would not cross-post their review requests on these portals, if the app ratings would not change as desired by app developers.

Fourth, per app we compared the collected fake reviews to each other. We searched for apps that received fake reviews with a rating of 1–2 stars as well as fake reviews with a rating of 4–5 stars. These reviews are most likely written by reviewers that posted their honest opinion about an app, either positive or negative. This applies for only 32 of the 1,890 (1.69%) collected apps. For these apps, 41 1-star and 2-star reviews out of 8,607 reviews (0.48%) were provided.

Another limitation is that although we filtered fake reviews for reviews in English language only and targeted the US storefront of the Apple App Store, reviews in English language could have been submitted to other storefronts. This could be a possible reason why we were only able to identify 8,607 of the initially collected 60,431 fake reviews within the app store, i.e., the official reviews dataset. In this case, the moderation of reviews by app store operators is less strict than observed.

Further, review exchange portals are also used by app developers, since the reward of providing a fake review is a credit which can be redeemed into another fake review for an app specified. As a result, the amount of requirements-related information included in fake reviews could be influenced since some users of the portals might be app developers.

For in-app purchases, we were unable to receive the offers programatically. Therefore, we could not compare the manually collected in-app purchases for apps affected by fake reviews against other app store in-app purchases. Also, we could not identify statistics which we could have used alternatively, or statistics on the monetization through ads.

However, we decided to focus on the Apple App Store because of our prior experience with the technology and because this app store does not impose major API limitations to retrieve its data, e.g., compared to Google Play with limits the number of accessible reviews to 2,000 per app and uses captchas. To have reliable results for the Apple App Store itself we crawled the largest dataset of about 62 million app reviews which has been analyzed so far to our knowledge. Thereby, we also avoid the App Sampling Problem for app store mining, described by Martin et al. (2015).

The questionnaire we conducted with the paid review providers was hidden as a request for buying app reviews. We cannot assure that the responses only contain true statements. Therefore, we contacted several providers again after a few weeks using a different identity and communication channel, such as Skype. For providers we contacted again, their responses did not change.

When manually labelling data, such as when finding agreements for potential matches between reviews within the fake reviews dataset and official reviews dataset, we used two human annotators which independently solved the task. In case of mismatches (3%), we resolved the conflicts using a third human annotator. However, single reviews could have been mismatched.

For statistical tests, in addition to reporting the p value we also calculated the effect size. For t-tests we calculated the effect size using Cohen's d, that is the difference between means divided by the pooled standard deviation (Cohen 1988). For Wilcoxon tests we calculated the r value, dividing the z distribution by the square root of the number of samples (Fritz et al. 2012). We report the effect size considering the following values, for Cohen's d (0.2 = small, 0.5 = medium, 0.8 = large) and for the correlation coefficient r (0.10 = small, 0.30 = medium, 0.50 = large). In two cases, although a statistical difference was observed the effect size revealed that the magnitude between differences is near zero. For these, the tests need to be repeated with additional samples (i.e., fake reviews) to show a statistical difference.

We want to stress that our classifier is only a first attempt to automatically identify fake reviews and not the main contribution of our paper. We wanted to verify if the features identified in our study are relevant for identifying fake reviews.

The machine learning model could be overfitted. This may be due to the small amount of fake reviews. More fake reviews need to be collected to improve the results. We tried to minimize overfitting by using k-fold cross validation. To minimize the impact of randomly chosen data, we used another 8,000 randomly selected official reviews and were able to reproduce our results reported in the paper.

Moreover, we cannot ensure that all official reviews are non-fake reviews. As stated before, to provide a gold-standard dataset for regular reviews as well, all fake reviews must be known and removed which is the problem we are trying to solve in this paper. For this reason, we optimized the classifier for precision only.

We leave the development of an advanced classifier for future research. Additional features, e.g., the emotion of users (Calefato et al. 2017; Martens and Johann 2017), have to be analyzed to strengthen the results. For that we publicly share our gold-standard fake reviews dataset within our replication package.

7 Related Work

User reviews are a valuable resource for decision making—both to other users and developers. To our best knowledge no published studies on fake reviews for software products exist.

A similar phenomena to fake reviews for software products, called web spamming, has been studied earlier when web pages began to compete for the rank within search engines' results. This is comparable to apps competing within app stores today. Web spamming is defined as the act of misleading search engines to rank pages higher than they deserve. Website operators edit their pages, e.g., by repeatedly adding specific terms that improve their ranking in search results (Gyongyi and Garcia-Molina 2005). Based on this a definition for user-generated content emerged, called opinion spam. The authors divide opinion spam into three categories, of which the first category are untruthful opinions. These mislead readers and opinion mining systems by giving undeserved/unjust either positive reviews to promote or negative reviews to damage the reputation of a target object. Untruthful opinions are also commonly known as fake reviews (Jindal and Liu 2008). This definition has been refined several times, e.g., by adding that fake reviews are written by persons as if they were real customers (Ott et al. 2011).

Recent research in other areas studied fake reviews, e.g., for products sold on Amazon, hotels rated on TripAdvisor, and businesses rated on Yelp. Jindal and Liu (2008) first analyzed opinion spam. The authors analyze 5.8 million reviews and 2.14 million reviewers from Amazon to detect spam activities and present techniques to detect those. Due to the difficulty to create a fake reviews dataset, the authors used duplicate and near-duplicate reviews written by the same reviewers on different products. In our work we were able to extract data from fake review providers to achieve more reliable results.

Ott et al. (2011) state that an increasing amount of user reviews is provided. Due to their value, platforms containing user reviews are becoming targets of opinion spam for potential monetary gain—our work confirms this for app stores and provides further insight on the fake review offers and policies. The authors focus on analyzing deceptive fake reviews, which are reviews that have been written to sound authentic, instead of disruptive fake reviews. The authors highlighted that there are few sources for deceptive fake reviews. To overcome the issue they hired 400 humans using a crowd-sourcing platform to write fake

hotel reviews. Their classifier integrates work from psychology and computational linguistics. It has an accuracy of nearly 90% on the crowdsourced dataset. The authors showed that classifiers are better in recognizing deceptive fake reviews compared to humans which scored an accuracy of 61% at most.

Feng et al. (2012) analyzed fake reviews on TripAdvisor and Amazon. They identified fake reviews based on the hypothesis that for a given domain a representative distribution of review rating scores exist which is distorted by fake reviews. The authors used an unsupervised learning approach to create a review dataset that is labeled automatically based on rating distributions. Using a statistical classifier trained on that dataset the authors were able to detect fake reviews with an accuracy of 72%.

Mukherjee et al. (2013b), compared to existing studies, used real fake reviews instead of pseudo-fake reviews, e.g., generated using crowdsourcing platforms. Their dataset consists of fake reviews published on Yelp, filtered and marked by the platform itself. The authors used the supervised approach of Ott et al. (2011) on their dataset an achieved a significantly lower accuracy of 67.8%, compared to 89.6%. The authors found that the word distribution of pseudo-fake reviews is different to the word distribution of real reviews. However, this does not apply for fake reviews within their dataset. Instead of using linguistic features, the authors suggest to use behavioral features. These include numeric values, such as the maximum number of reviews of a reviewer per day, or the review length. We followed this approach when designing our classifier and achieved encouraging results of up to 97% precision.

Fake reviews have also been frequently discussed within the media. Streitfeld (Times 2017) reported that every fifth review submitted to Yelp is detected as dubious by its internal filters. Instead of removing dubious reviews, these are moved to the second page where they are read by less users. As fake reviews further increase, Yelp began a 'sting' campaign to publicly expose businesses buying fake reviews.

8 Conclusion

App reviews can be a valuable, unique source of information for software engineering teams reflecting the opinions and needs of actual users. Also potential users read through the reviews before deciding to download an app, similar to buying other products on the Internet. Our work shows that part of app reviews in app stores are fake—that is, they are incentivized and might not reflect spontaneous, unbiased opinions.

We analyzed the market of fake review providers and their fake reviewing strategies and found that developers buy reviews to relatively expensive prices of a few dollars or deal with reviews in exchange portals. Fake reviews are written to look authentic and are hard to recognize by humans. We identified differences between fake and official reviews. We found that properties of the corresponding app and reviewer are most useful to determine if a review is fake. Based on the identified differences, we developed, trained, fine-tuned, and compared multiple supervised machine learning approaches. We found that the Random Forest classifier identifies fake reviews, given a proportional distribution of fake and regular reviews as reported in other domains, with a recall of 91% and AUC/ROC value of 98%. We publicly share our gold-standard fake reviews dataset to enable the development of more accurate classifiers to identify fake reviews. Our work helps app store mining researchers to sample apps and perform data cleaning to achieve more reliable results. Further, tools for app users and store operators can be built based on our findings to detect if app reviews are trustworthy and to take further actions against fake reviewers.

Acknowledgment This research was partially supported by the European Union Horizon 2020 project OpenReq under grant agreement no. 732463.

References

9to5Mac (2017) Bell faces $1.25M fine for posting fake reviews of its app in the App Store. https://9to5mac.com/2015/10/14/bell-fake-app-store-reviews/

Aizerman MA, Braverman EA, Rozonoer L (1964) Theoretical foundations of the potential function method in pattern recognition learning. In: Automation and Remote Control, no. 25 in Automation and Remote Control, pp 821–837

AppAnnie (2017) App Annie. https://www.appannie.com/en/

Apple (2017) App Store Review Guidelines. https://developer.apple.com/app-store/review/guidelines/

Bird S, Klein E, Loper E (2009) Natural language processing with python, 1st edn. O'Reilly Media, Inc

Calefato F, Lanubile F, Novielli N (2017) Emotxt: a toolkit for emotion recognition from text. arXiv:170803892

Carreno LVG, Winbladh K (2013) Analysis of user comments: an approach for software requirements evolution. In: 2013 35th international conference on software engineering (ICSE), pp 582–591. https://doi.org/10.1109/ICSE.2013.6606604

Chawla NV, Japkowicz N, Kotcz A (2004) Special issue on learning from imbalanced data sets. ACM Sigkdd Explor Newsl 6(1):1–6

Chen N, Lin J, Hoi SCH, Xiao X, Zhang B (2014) Ar-miner: mining informative reviews for developers from mobile app marketplace. In: Proceedings of the 36th international conference on software engineering. ICSE 2014. ACM, New York, pp 767–778. https://doi.org/10.1145/2568225.2568263. http://doi.acm.org/10.1145/2568225.2568263

Cohen J (1988) Statistical power analysis for the behavioral sciences. Lawrence Erlbaum Associates, Hillsdale

Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297. https://doi.org/10.1007/BF00994018

Dickerson JP, Kagan V, Subrahmanian VS (2014) Using sentiment to detect bots on twitter: Are humans more opinionated than bots? In: 2014 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM 2014), pp 620–627. https://doi.org/10.1109/ASONAM.2014.6921650

DigitalTrends (2018) Can you really trust app store ratings? We asked the experts. https://www.digitaltrends.com/android/can-you-really-trust-app-store-ratings/

Drummond C, Holte RC et al (2003) C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: Workshop on learning from imbalanced datasets II, Citeseer, vol 11, pp 1–8

Fakespot (2017) Fakespot. http://fakespot.com

Feng S, Xing L, Gogar A, Choi Y (2012) Distributional footprints of deceptive product reviews. ICWSM 12:98–105

Ferrara E, Varol O, Davis CA, Menczer F, Flammini A (2014) The rise of social bots. CoRR arXiv:1407.5225

Finkelstein A, Harman M, Jia Y, Martin W, Sarro F, Zhang Y (2017) Investigating the relationship between price, rating, and popularity in the blackberry world app store. Inf Softw Technol 87:119–139. https://doi.org/10.1016/j.infsof.2017.03.002. http://www.sciencedirect.com/science/article/pii/S095058491730215X

Fritz CO, Morris PE, Richler JJ (2012) Effect size estimates: current use, calculations, and interpretation. J Exp Psychol Gen 141(1):2

Fu B, Lin J, Li L, Faloutsos C, Hong J, Sadeh N (2013) Why people hate your app: making sense of user feedback in a mobile app store. ACM, Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1276–1284

Gao C, Xu H, Hu J, Zhou Y (2015) Ar-tracker: track the dynamics of mobile apps via user review mining. In: 2015 IEEE symposium on service-oriented system engineering (SOSE). IEEE, pp 284–290

Google (2017) Google play user ratings, reviews, and installs. https://play.google.com/about/storelisting-promotional/ratings-reviews-installs/

Google (2018) Google Play Prohibited and Restricted Content. https://support.google.com/contributionpolicy/answer/7400114

Google (2019) In reviews we trust - making Google Play ratings and reviews more trustworthy. https://android-developers.googleblog.com/2018/12/in-reviews-we-trust-making-google-play.html

Guzman E, Maalej W (2014) How do users like this feature? In: A fine grained sentiment analysis of app reviews. 2014 IEEE 22nd international requirements engineering conference (RE), pp 153–162. https://doi.org/10.1109/RE.2014.6912257

Gyongyi Z, Garcia-Molina H (2005) Web spam taxonomy. In: First international workshop on adversarial information retrieval on the web (AIRWeb 2005)

Harman M, Jia Y, Zhang Y (2012) App store mining and analysis: MSR for app stores. In: 2012 9th IEEE working conference on mining software repositories (MSR 2012). IEEE, pp 108–111. https://doi.org/10.1109/MSR.2012.6224306. http://ieeexplore.ieee.org/document/6224306/

Ho TK (1995) Random decision forests. In: Proceedings of the third international conference on document analysis and recognition, vol 1. ICDAR '95. IEEE Computer Society, Washington, DC, p 278. http://dl.acm.org/citation.cfm?id=844379.844681

Hoon L, Vasa R, Schneider JG, Grundy J et al (2013) An analysis of the mobile app review landscape: trends and implications. In: Faculty of Information and Communication Technologies, Swinburne University of Technology, Tech Rep

Iacob C, Harrison R (2013) Retrieving and analyzing mobile apps feature requests from online reviews. In: 2013 10th working conference on mining software repositories (MSR), pp 41–44. https://doi.org/10.1109/MSR.2013.6624001

Iacob C, Veerappa V, Harrison R (2013) What are you complaining about?: A study of online reviews of mobile applications In: Proceedings of the 27th international BCS human computer interaction conference, BCS-HCI '13. British Computer Society, Swinton, pp 29:1–29:6. http://dl.acm.org/citation.cfm?id=2578048.2578086

Iacob C, Harrison R, Faily S (2014) Online reviews as first class artifacts in mobile app development. In: Memmi G, Blanke U (eds) Mobile computing, applications, and services. Springer International Publishing, Cham, pp 47–53

Jindal N, Liu B (2008) Opinion spam and analysis. In: Proceedings of the 2008 international conference on web search and data mining, WSDM '08. ACM, New York, pp 219–230. https://doi.org/10.1145/1341531.1341560. http://doi.acm.org/10.1145/1341531.1341560

Johann T, Stanik C, AMA B, Maalej W (2017) Safe: a simple approach for feature extraction from app descriptions and app reviews. In: 2017 IEEE 25th international requirements engineering conference (RE), pp 21–30. https://doi.org/10.1109/RE.2017.71

Khalid H (2013) On identifying user complaints of ios apps. In: Proceedings of the 2013 international conference on software engineering, ICSE '13. IEEE Press, Piscataway, pp 1474–1476. http://dl.acm.org/citation.cfm?id=2486788.2487044

Lee K, Caverlee J, Webb S (2010) Uncovering social spammers: Social honeypots + machine learning. In: Proceedings of the 33rd international ACM SIGIR conference on research and development in information retrieval, SIGIR '10. ACM, New York, pp 435–442. https://doi.org/10.1145/1835449.1835522. http://doi.acm.org/10.1145/1835449.1835522

Luca M, Zervas G (2016) Fake it till you make it: reputation, competition, and yelp review fraud. Manag Sci 62(12):3412–3427

Maalej W, Kurtanović Z, Nabil H, Stanik C (2016a) On the automatic classification of app reviews. Requir Eng 21(3):311–331. https://doi.org/10.1007/s00766-016-0251-9

Maalej W, Nayebi M, Johann T, Ruhe G (2016b) Toward data-driven requirements engineering. IEEE Softw 33(1):48–54. https://doi.org/10.1109/MS.2015.153

Martens D, Johann T (2017) On the emotion of users in app reviews. In: Proceedings of the 2nd international workshop on emotion awareness in software engineering, SEmotion '17. IEEE Press, Piscataway, pp 8–14. https://doi.org/10.1109/SEmotion.2017.6

Martin W, Harman M, Jia Y, Sarro F, Zhang Y (2015) The app sampling problem for app store mining. In: Proceedings of the 12th working conference on mining software repositories, MSR '15. IEEE Press, Piscataway, pp 123–133. http://dl.acm.org/citation.cfm?id=2820518.2820535

Martin W, Sarro F, Jia Y, Zhang Y, Harman M (2016) A survey of app store analysis for software engineering. IEEE Trans Softw Eng PP(99):1–1. https://doi.org/10.1109/TSE.2016.2630689. http://ieeexplore.ieee.org/document/7765038/

Mukherjee A, Venkataraman V, Liu B (2013a) Fake review detection: classification and analysis of real and pseudo reviews

Mukherjee A, Venkataraman V, Liu B, Glance NS (2013b) What yelp fake review filter might be doing? In: ICWSM

Nayebi M, Ruhe G (2017) Optimized functionality for super mobile apps. In: 2017 IEEE 25th international requirements engineering conference (RE), pp 388–393. https://doi.org/10.1109/RE.2017.72

Nayebi M, Marbouti M, Quapp R, Maurer F, Ruhe G (2017) Crowdsourced exploration of mobile app features: a case study of the fort mcmurray wildfire. In: 2017 IEEE/ACM 39th international conference on software engineering: software engineering in society track (ICSE-SEIS), pp 57–66. https://doi.org/10.1109/ICSE-SEIS.2017.8

Oh J, Kim D, Lee U, Lee JG, Song J (2013) Facilitating developer-user interactions with mobile app review digests. In: CHI '13 extended abstracts on human factors in computing systems, CHI EA '13. ACM, New York, pp 1809–1814. https://doi.org/10.1145/2468356.2468681. http://doi.acm.org/10.1145/2468356.2468681

Ott M, Choi Y, Cardie C, Hancock JT (2011) Finding deceptive opinion spam by any stretch of the imagination. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies - volume 1, HLT '11. Association for Computational Linguistics, Stroudsburg, pp 309–319. http://dl.acm.org/citation.cfm?id=2002472.2002512

Pagano D, Brügge B (2013) User involvement in software evolution practice: a case study. In: Proceedings of the 2013 international conference on software engineering, ICSE '13. IEEE Press, Piscataway, pp 953–962. http://dl.acm.org/citation.cfm?id=2486788.2486920

Pagano D, Maalej W (2013) User feedback in the appstore—an empirical study, pp 125–134. https://doi.org/10.1109/RE.2013.6636712. http://ieeexplore.ieee.org/document/6636712/

Palomba F, Linares-Vásquez M, Bavota G, Oliveto R, Penta MD (2015) Poshyvanyk D. In: 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME). Lucia AD, User reviews matter! tracking crowdsourced reviews to support evolution of successful apps, pp 291–300. https://doi.org/10.1109/ICSM.2015.7332475

Panichella S, Sorbo AD, Guzman E, Visaggio CA, Canfora G, Gall HC (2015) How can I improve my app? Classifying user reviews for software maintenance and evolution. In: 2015 IEEE international conference on software maintenance and evolution (ICSME), pp 281–290. https://doi.org/10.1109/ICSM.2015.7332474

Park D, Sachar S, Diakopoulos N, Elmqvist N (2016) Supporting comment moderators in identifying high quality online news comments. In: Proceedings of the 2016 CHI conference on human factors in computing systems, CHI '16. ACM, New York, pp 1114–1125. https://doi.org/10.1145/2858036.2858389. http://doi.acm.org/10.1145/2858036.2858389

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830

Saito H, Toyoda M, Kitsuregawa M, Aihara K (2007) A large-scale study of link spam detection by graph algorithms. ACM, Proceedings of the 3rd international workshop on adversarial information retrieval on the web, pp 45–48

SciKit (2018) SciKit Learn: 4.3. Preprocessing data. http://scikit-learn.org/stable/modules/preprocessing.html

Seneviratne S, Seneviratne A, Kaafar MA, Mahanti A, Mohapatra P (2017) Spam mobile apps: characteristics, detection, and in the wild analysis. ACM Trans Web 11(1):4:1–4:29. https://doi.org/10.1145/3007901. http://doi.acm.org/10.1145/3007901

Stringhini G, Wang G, Egele M, Kruegel C, Vigna G, Zheng H, Zhao BY (2013) Follow the green: growth and dynamics in twitter follower markets. In: Proceedings of the 2013 conference on internet measurement conference, IMC '13. ACM, New York, pp 163–176. https://doi.org/10.1145/2504730.2504731. http://doi.acm.org/10.1145/2504730.2504731

Subrahmanian VS, Azaria A, Durst S, Kagan V, Galstyan A, Lerman K, Zhu L, Ferrara E, Flammini A, Menczer F, Waltzman R, Stevens A, Dekhtyar A, Gao S, Hogg T, Kooti F, Liu Y, Varol O, Shiralkar P, Vydiswaran VGV, Mei Q, Huang T (2016) The DARPA twitter bot challenge. CoRR arXiv:1601.05140

Sussin J, Thompson E (2018) The consequences of fake fans, likes and reviews on social networks. https://www.gartner.com/doc/2091515/consequences-fake-fans-likes-reviews

Svedic Z (2015) The effect of informational signals on mobile apps sales ranks across the globe

Times NY (2017) Buy reviews on yelp, get Black Mark. http://www.nytimes.com/2012/10/18/technology/yelp-tries-to-halt-deceptive-reviews.html

Torgo L (2010) Data mining with R: learning with case studies, 1st edn. Chapman & Hall/CRC, Boca Raton

Villarroel L, Bavota G, Russo B, Oliveto R, Penta MD (2016) Release planning of mobile apps based on user reviews. In: 2016 IEEE/ACM 38th international conference on software engineering (ICSE), pp 14–24. https://doi.org/10.1145/2884781.2884818

Wang S, Minku LL, Yao X (2018) A systematic study of online class imbalance learning with concept drift. IEEE Trans Neural Netw Learn Syst 29(10):4802–4821. https://doi.org/10.1109/TNNLS.2017.2771290

Daniel Martens received the B.Sc. and M.Sc. degree in Computer Science from the University of Hamburg. He is a Ph.D. candidate in the Applied Software Technology group at the University of Hamburg. His research interests include user feedback, data-driven software engineering, context-aware adaptive systems, crowdsourcing, and mobile computing. Besides his academic career, Daniel Martens also worked as a software engineer where he developed more than 30 top-rated iOS applications. He is a student member of the IEEE, ACM, and German Computer Science Society (GI). The photo of Daniel Martens was taken by UHH/Sukhina.

Walid Maalej is a professor of informatics at the University of Hamburg where he leads the Applied Software Technology group. His research interests include user feedback, data-driven software engineering, context-aware adaptive systems, e-participation and crowdsourcing, and software engineering's impact on society. He received his Ph.D. in software engineering from the Technical University of Munich. He is currently a steering committee member of the IEEE Requirements Engineering conference and a Junior Fellow of the German Computer Science Society (GI). The photo of Prof. Walid Maalej was taken by Köpcke-Fotografie.