2023-05

A Gradient Boosted ML Approach to Feature Selection for Wireless Intrusion Detection

Mondal, Birupaxha

Independent University, Bangladesh

A Gradient Boosted ML Approach to Feature Selection for Wireless Intrusion Detection

Birupaxha Mondal, Fahim Faisal, Zeba Tusnia Towshi, Md Fahad Monir, and Tarem Ahmed
Department of Computer Science and Engineering
Independent University, Bangladesh (IUB)
{1830896, 1831123, 1810222, fahad.monir, tarem}@iub.edu.bd

***Abstract*—The proliferation of Wi-Fi-enabled devices makes security a non-negotiable part of connectivity. As new attacks are discovered that compromise the security of devices in the wireless ecosystem, it is becoming increasingly crucial for intrusion detection systems to generalize to these novel attacks. Machine Learning gives us an approach to do that. In this paper, we provide a feature elimination technique to narrow down the set of features necessary to build such an ML-based solution that takes into account possible class imbalance issues in intrusion datasets. With features extracted using this technique from the AWID dataset, we use a gradient-boosted model to show that these features are necessary to generalize to new attack types in the AWID test dataset.**

***Index Terms*—Intrusion Detection, Machine Learning, Wi-Fi, Wireless Security, AWID**

I. Introduction

Wireless local area networks (WLANs) have become an integral part of our daily lives and are widely used in various settings including homes, offices, and public spaces. In 2022, 351 exabytes of monthly internet traffic were anticipated to be generated globally, with Wi-Fi networks contributing 57% [1]. Compared to 169 million hotspots in 2018, there will be roughly 628 million public Wi-Fi hotspots worldwide by 2023 [2]. However, using Wi-Fi networks also increases the risk of security breaches and unauthorized access. Kaspersky Lab reported that about 28% of Wi-Fi hotspots worldwide were unsecured, with the risk of data breaches of users [3].

To address the security issues, intrusion detection systems (IDSs) have been proposed to detect and prevent malicious activities in WLANs. An IDS is a security tool that detects any activity violating networked systems' security standards. As the nature of technology and attack strategies are constantly evolving, the IDS must be flexible and adaptable to counter new attacks. To discriminate between legitimate activity and intrusions, a machine learning-based Wi-Fi network intrusion detection system is an innovative approach to address this challenge. However, ML-based solutions haven't performed well in this field due to the class imbalance issue [4]. In datasets, the amount of intrusive or anomalous data samples is so low that it is difficult to train a model that reliably results in low false-positive and false-negative rates, both of which are important for an intrusion detection system.

The AWID dataset, which consists of Wi-Fi packet traffic, is used in this study to investigate intrusion detection. We use a gradient-boosted ML model and apply a modified feature selection algorithm to select a set of features and show that they are necessary for generalizing to unseen attack classes in the AWID test dataset.

The rest of this paper is organized as follows. Section II summarizes our contributions in light of present literature. Section III presents our proposed framework. Section IV analyses our results on the benchmark AWID dataset. Section V concludes with a discussion of future possibilities.

II. Related Work and Our Contribution

Kolias et al. released the AWID dataset which contains real traces of both normal and intrusive 802.11 traffic [5]. With all 154 features in the dataset, their best accuracy of trained model was 96.20% using the J48 classifier, and 96.26% when the number of features was reduced from 154 to 20 [5]. Udaya et al. implemented multiple ML algorithms with 111, 41, and 10 features acquired from various feature selection methods to increase the detection accuracy on the AWID-CLS-R dataset from 92.17% to 95.12% [6]. In [7], Wang et al. reduced the feature set to 71 features, eliminating features with zero variance and features with missing values and trained a DNN-based machine learning model. For 4-class classification, they had an average accuracy of 92.49%. In contrast, we have achieved an overall test accuracy 99% after a first run through LightGBM using all of our features.

Our features selection approach is extremely fast and addresses the dataset's class imbalance when establishing the features' importance. The following are the contributions made in this work:

1) The use of a novel feature selection technique that considers the AWID dataset's class imbalance and helps us narrow down a list of features that are important in detecting the categories.
2) The list of selected features is shown to be necessary for generalizing to the specific attacks which are not present in the training dataset.
3) We achieved a high detection rate for flooding attacks, and an impersonation detection performance that is competitive with recent deep learning methods, all with a lower order of magnitude training time.

This work will contribute to the development of effective and efficient intrusion detection systems for Wi-Fi networks.

III. Proposed Framework and Testbed

A. Dataset Overview

For this paper, we use the Aegean Wi-Fi Intrusion Dataset (AWID) that was made public in 2014 by Kolias *et. al.* [5] [9]. There are two differentiating factors within the four different datasets they offer.

The first is the number of class labels. The `ATK` variant labels each sample with either 'normal' or the name of the exact attack that it was a part of. The `CLS` variant reduces the number of classes to four: 'normal', 'impersonation', 'injection', and 'flooding'. The second differentiating factor is the size of the dataset. The full dataset is denoted by F, and the reduced dataset is denoted by R.

We will be using the reduced dataset with four categories. The distribution of each of the categories across the training and test set is as follows:

As we can see, the class distribution is highly imbalanced, and this issue stems from the nature of network traffic. Normal internet traffic constitutes a significant portion of internet traffic worldwide, and any trace of such traffic flow is bound to have an imbalance of this sort. This imbalance will come into play in our subsequent analysis as it adversely affects the accuracy of machine learning models.

B. Preprocessing

There are 154 features in the AWID dataset. However, a lot of these features have values missing in the majority of samples. Specifically, management frames constitute a small portion of the overall network trace, so all the fields corresponding to management frames are empty for the majority of the samples. We throw away all the features that are missing the majority of their values.

Our motivation for doing so was two-fold. First, it is not easy for a ML-model to observe patterns among features when most of the values are missing. Second, given the amount of missing data, there is no straightforward method to impute the values without introducing bias. [5] used a number of models to evaluate their performance on the whole feature set and a reduced set consisting of 20 features. They found very little difference in accuracy. We also eliminated all features with constant values.

We further eliminated the following features: *wlan.ra*, *wlan.da*, *wlan.ta*, *wlan.sa* and *wlan.bssid*. The domain of these features is the space of MAC addresses. We found no easy way to encode the information within them. We also eliminated all time-related features except *frame.time_delta*, since they are tied closely to the AWID experiment and are unlikely to generalize. The rest of the features with missing values were imputed with the most-frequently occurring values.

TABLE I: Class distribution in AWID dataset

	AWID-CLS-R-Trn	AWID-CLS-R-Tst
Normal	1,633,190	530,785
Impersonation	48,522	20,079
Injection	65,379	16,682
Flooding	48,484	8,079
Total	1,795,575	575,625

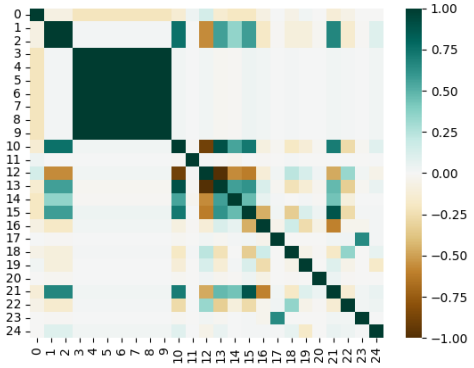


Fig. 1: Correlation matrix of dataset. Large square blocks represent groups of features with the same values.

C. Initial Feature Selection

Following the steps in III-B, we calculated the correlation matrix of the dataset to eliminate perfectly correlated features. The correlation matrix is visualized in Fig. 1. Note that this matrix left out *wlan.fc.ds*.

From the matrix, we can see that certain groups of features are highly correlated. The values of the features in each group were identical in every sample. From each of these sets of features, we only kept one feature, and eliminated the rest.

Next, among the remaining features, we identified three categorical variables that can take on multiple values. Specifically, *wlan.fc.ds* can take on 4 values (0–4) (but the presence of only 3 were found on the entire dataset), *wlan.fc.type* can take on 3 values (0–2), and *wlan.fc.subtype* can take on 14 different values (0–13). We one-hot encoded these features. By their very nature, machine learning models try to identify significance in the magnitude of numbers. Since no such significance exists within categorical features, it's best to replace a variable with N possible values with N binary variables that show the presence/absence of each value. This also allows us to narrow down the set of features to certain values that each of these categorical values take.

D. Model Choice and Feature Elimination Technique

For this work, we adopt LightGBM as our main vehicle for machine learning tasks. LightGBM belongs to a class of popular models that employ Gradient Boosting. Gradient boosting uses an ensemble of *weak learners*, usually decision trees. At a high-level, it works like this: a weak learner learns some weak (not too accurate) function for the problem at hand. Each subsequent weak learner tries to amend the samples that its previous weak learner got wrong. In this way, intuitively, each weak learner moves a little bit towards the direction of negative gradient of some cost function.

While our model of choice is able to report an explicit feature importance score, their computation method naturally

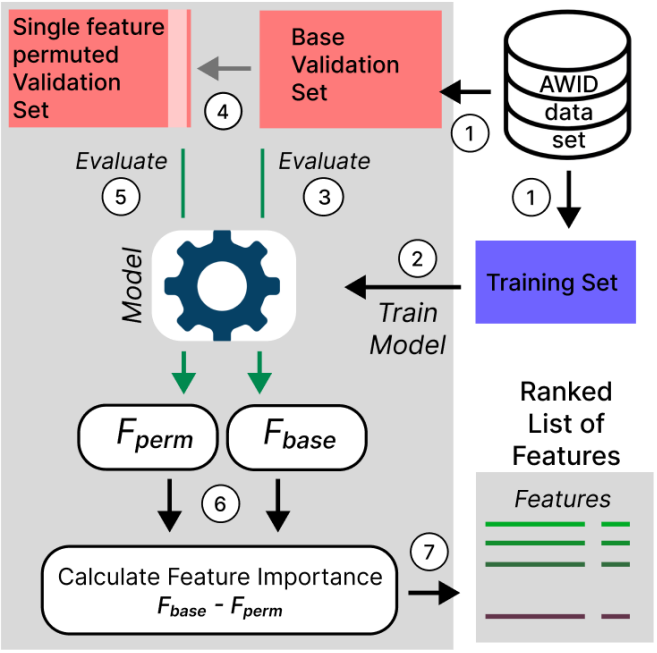


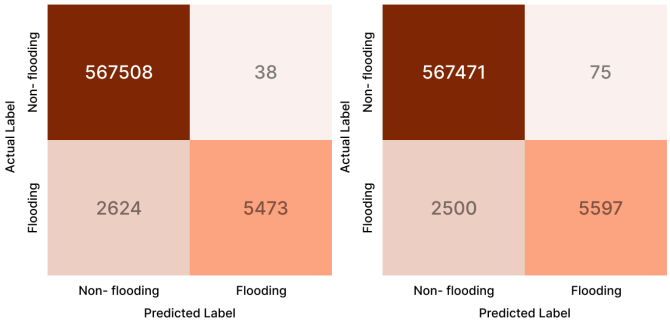
Fig. 2: Diagram of feature selection technique. Steps 4, 5, and 6 are repeated for every feature.

assigns higher importance to continuous features or categorical features with a large number of possible values, because they appear more often in the branches of decision trees. Therefore, for feature elimination, we use *permutation importance*.

Permutation importance works as follows: we first train a model on the training dataset and calculate an accuracy metric A on the validation dataset. Then, we separately permute each of the features of the validation dataset and evaluate the model's accuracy on this transformed validation set to get A_{perm}. The difference between the two accuracy scores gives us a sense of the feature's importance. For our purposes, we chose to use the **F1-score** as the metric for permutation importance. This is because of the class-imbalance issue in the AWID dataset. Because the number of intrusive instances are small, differences in accuracy will be small. By calculating F1-score by treating the intrusive class as the positive class, we heavily penalize those features that cause any appreciable level of degradation in the intrusion class detection. Algorithm 1 lists the recipe for feature importance ranking concretely, and the whole process is visualized in Fig. 2.

Algorithm 1 Feature Ranking Procedure

```
Initialize non-overlapping training and validation datasets
Train a model with the training dataset
Calculate baseline F1-score on validation set, F_base
for each feature of the dataset do
    Permute the values of the feature in validation dataset
    Establish F1-score of this validation dataset, F_perm
    Importance of feature: F_base − F_perm
end for
```



(a) Full Dataset (b) Balanced Dataset

Fig. 3: Flooding detection trained on different sized datasets. Balanced dataset training improves flooding detection.

IV. Results and Analyses

A. Flooding Detection

In order to look at the flooding case more generally, we grouped each of the labels 'normal', 'impersonation', and 'injection' into a more general 'non-flooding' category. We then trained our model for the binary classification of 'non-flooding' and 'flooding' examples.

When trained on the entire training set, LightBGM achieved an accuracy of 99.5% and an F1 score of 0.8, which is similar to most of the approaches explored for detecting it. Fig. 3a show these results. Similar to this model, most approaches also report low false-positive rates. This was a strong indicator that most models trained on this dataset were far more adept and *confident* in their detection of non-flooding samples when the actual category is also non-flooding. We also suspected that this confidence on detecting actual non-flooding cases stems from the class imbalance in the dataset.

To assess the impact of training on a balanced dataset, we used a reduced sample of the training set, which contained equal numbers of 'non-flooding' and 'flooding' examples. The confusion matrix of this trained model is shown on Fig. 3b. While not by a large margin, both the accuracy and F1 score are higher for this model, even though its training set was an order of magnitude smaller than the full training set.

To show the confidence level changes of the model in identifying each of the types, we analyzed the distribution of the predicted probabilities of the classes by our model. Tabel II shows the mean predicted probabilities. This confirms when a test sample is actually non-flooding, it identifies it as such with

TABLE II: Predicted Class Probabilities

	Training on full training dataset		Training on balanced subset	
	Reported Non-Flooding	Reported Flooding	Reported Non-Flooding	Reported Flooding
Actual Non-Flooding	0.9995	0.0005	0.9993	0.0007
Actual Flooding	0.333	0.667	0.306	0.694

a lot more confidence than it does when reporting a flooding sample as flooding. We also observed that after training with the reduced, balanced training set, its true-class distinction confidence on flooding samples increased by 3 percentage points. This suggests that having more flooding examples should be a viable way of increasing detection accuracy.

Next, we applied our feature elimination algorithm after training on the small, balanced dataset. Table III shows the 9 features we shortlisted based on the largest drop in F1 score on an unbalanced validation set sampled from the training set (mutually exclusive with the balanced training sample).

We trained our model again with these 9 features and got performance similar to those of the model trained with balanced training set in Fig. 3a, Fig. 3b and Table II. The accuracy for flooding dataset remained at 99.5% and we got an F1 score of 0.81.

The 'Normal Probability Split' columns in Table IV show a cross-examination of the results with the labels in the ATK dataset. The specific categories that the model missed, namely *cts*, *power_saving*, *probe_request*, and *rts*, were all missing in the training dataset. Therefore, given the feature set, it would seem that gradient-boosted models do not generalize to flooding attack types that it hasn't seen before during training.

Normally, a model reports a class in a classification problem if its predicted probability is higher than that of the other classes. Because of the lower confidence for 'flooding' cases reported in Table II, we extracted another set of predictions from the same model where we classified a sample as 'flooding' if it's predicted probability was above 1%, and 'non-flooding' otherwise. The detection rate of 'flooding' type jumped to 95.4%, and the cross-examination results with the ATK dataset are shown in the 'Flooding-favored Probability Split' columns of Table IV. Here, we see that it detected almost all *cts* and *probe_request* types. However, this came at the cost of a high number of false positives.

B. Impersonation/Injection Detection

Detecting injection attacks from the AWID dataset has been a very easy task for ML models, largely because all the specific injection attacks present in the test dataset are also present in the training dataset. Impersonation attacks, on the other hand, have been difficult to classify. [5] obtained near-perfect accuracy for injection attacks, but did poorly in impersonation detection. Similarly, a *one-vs-all* classification of injection attacks using LightGBM achieves an F1-score of 0.99. But for impersonation detection, it achieves an F1-score of 0.13.

The authors in [8] suggest a two-phase intrusion detection scheme in which they first distinguish between the three classes 'normal', 'flooding', and the grouped class 'Impersonation/Injection'. If a sample is identified as 'Impersonation/Injection', it is fed into the second-stage ML model that differentiates between the two. The motivation for this approach came from their earlier work in [10], where they found that a Random Forest used for 4-class classification of the reduced dataset misclassified most impersonation examples as injections. While we couldn't replicate their results, we found that this approach implemented with LightGBM did offer a better classification scheme for the combined class 'Impersonation/Injection' than the alternative of binary classification for Impersonation alone.

A first pass through LightGBM with our entire feature set yielded an overall test accuracy of 98.5% and an F1-score for 'Impersonation/Injection' detection of 0.91. Fig. 4a shows the confusion matrix of this model. While the detection of flooding samples hasn't improved, we get better overall detection of 'Impersonation' under the guise of 'Impersonation/Injection'.

We then used the feature elimination technique to narrow down the number of important features to 16. These 16 shortlisted features, along with the drop in F1-score that they caused are listed in Table V.

Training our model again on this reduced feature set gives the confusion matrix on Fig. 4b. As we can see, the detection of 'Impersonation/Injection' did improve somewhat, but it also came at the cost of an increased number of 'normal' samples being flagged as malicious. The accuracy and F1-score remain very similar at 98.5% and 0.91, respectively.

Table VI shows the cross examination with ATK dataset.

TABLE III: Selected features for Flooding

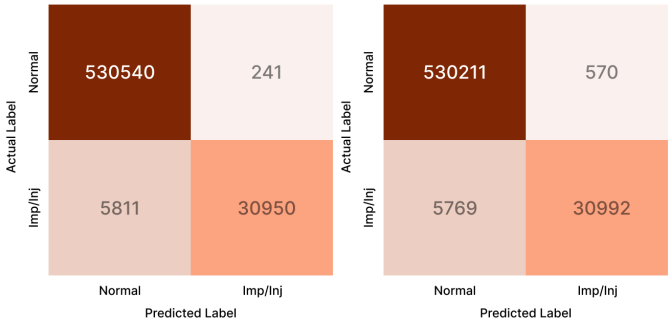
Feature	Drop in F1 score
frame.time_delta	3.22
frame.len	40.59
radiotap.dbm_antsignal	7.89
wlan.fc.retry	11.63
wlan.duration	26.85
wlan.seq	15.02
wlan.fc.type_0	83.7
wlan.fc.subtype_4	1.45
wlan.fc.subtype_10	0.06

TABLE IV: Flooding Types Missed

	Normal Probability Split		*Flooding-favored Probability Split*	
	Detected	**Missed**	**Detected**	**Missed**
amok	477	0	477	0
beacon	565	34	599	0
cts	0	1759	1756	3
deauthentication	4441	4	4444	1
dissociation	78	6	84	0
power_saving	0	165	0	165
probe_request	0	369	367	2
rts	0	199	0	199

TABLE V: Selected Features for Impersonation & Injection

Feature	Drop in F1 score	Feature	Drop in F1 score
frame.time_delta	9.82	wlan.duration	28.07
frame.len	47.52	wlan.frag	10.86
radiotap.datarate	19.68	wlan.seq	2.03
radiotap.channel.type.cck	7.28	wlan.fc.ds_0x01	6.47
radiotap.dbm_antsignal	10.75	wlan.fc.type_1	0.25
wlan.fc.frag	4.36	wlan.fc.subtype_0	54.9
wlan.fc.retry	1.68	wlan.fc.subtype_4	0.34
wlan.fc.protected	12.38	wlan.fc.subtype_5	0.19



(a) Full Feature Set (b) Reduced Feature Set

Fig. 4: Three class detection on different sized feature sets. Reduced feature set improves detection of intrusion class.

The major reason why *impersonation* detection results are usually bad is because simple *one-vs-all* classification fails to pick up *hirte* attacks, a type missing from the training dataset but constituting a major portion of the test dataset. However, the first-stage three-class classification suggested in [8] results in much better generalization to the unseen 'hirte' attack type, even though it sacrifices detection capacity of *chop_chop* attacks.

Since injection detection has been an easy problem for machine learning models, it is easy to differentiate between 'injection' and 'impersonation' following the three-class model above. The two-stage method somewhat dilutes the performance we can get for injection classification alone. Therefore, we propose the parallel use of two models. What these models are, and the procedure to use them are listed in Algorithm 2.

This procedure lets us keep the perfect detectibility of injection attacks (covering the chop_chop attacks missed by our three-class classifier), while simultaneously achieving the higher detection rate of impersonation attacks lent by the three-class classification. Following this, we achieve a detection rate for impersonation of 83.6%.

V. Conclusions and Future Work

In this paper, we employed gradient boosting to explore intrusion detectability in the AWID dataset. In particular, we used a novel feature elimination technique to select a small subset of features from the original feature set that is able to detect a large number of attacks against Wi-Fi devices. We also showed that these features are necessary, if not sufficient, in generalizing to new attack classes.

Further work on this front can explore incorporating other features either extracted from the ones we selected or chosen from the ones we didn't use to improve upon the classification confidence of machine learning models. In future work, we will explore the effects of deep-learning-based feature extraction from our list of selected features as proposed in [11] and training from a larger, balanced dataset taken from the F type datasets, and produce a working implementation that can detect intrusive traffic on wireless devices.

TABLE VI: Impersonation & Injection Types Missed

	One-vs-All		*Three Class*	
	Detected	**Missed**	**Detected**	**Missed**
Injection				
arp	13644	0	13644	0
chop_chop	2869	2	379	2492
fragmentation	167	0	167	0
Impersonation				
cafe_latte	379	0	370	9
evil_twin	611	0	585	26
hirte	483	18606	15805	3284

Algorithm 2 Separating Impersonation and Injection

```
M1 ← One-vs-all Injection Classifier
M2 ← Three-class Classifier
if M1 predicts 'Injection' then
    Report as injection
else
    if M2 predicts 'Impersonation/Injection' then
        Report as Impersonation
    end if
end if
```

References

[1] Cisco. Cisco VNI Global—2022 Forecast Highlights. Available online: https://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-highlights/pdf/Global_2022_Forecast_Highlights.pdf

[2] Cisco Annual Internet Report (2018–2023) White Paper Available online: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (accessed on 5 February 2023).

[3] 1 in 4 Wi-Fi Hotspots Just Waiting to Be Hacked, Kaspersky Lab Stats Show. Available online: https://www.kaspersky.com/about/press-releases/2016_1-in-4-wi-fi-hotspots-just-waiting-to-be-hacked-kaspersky-lab-stats-show (accessed on 5 February 2023).

[4] Vrizlynn L. L. Thing. "IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach". In: (2017), pp. 1–6.

[5] C. Kolias, G. Kambourakis, A. Stavrou and S. Gritzalis, "Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset," in IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 184-208, 2016.

[6] Thanthrige, U.; Samarabandu, J.; Wang, X. Machine learning techniques for intrusion detection on public dataset. In Proceedings of the 2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Vancouver, BC, Canada, 15–18 May 2016; pp. 1–4.

[7] Wang, S.; Li, B.; Yang, M.; Yan, Z. Intrusion Detection for WiFi Network: A Deep Learning Approach. In Wireless Internet. WICON 2018. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering; Chen, J.L., Pang, A.C., Deng, D.J., Lin, C.C., Eds.; Springer: Cham, Switzerland, 2018; pp. 95–104.

[8] A. Reyes, Abel, Francisco D. Vaca, G. A. C. Aguayo, Quamar Niyaz, and Vijay Devabhaktuni, "A Machine Learning Based Two-Stage Wi-Fi Network Intrusion Detection System" Electronics 9, no. 10: 1689, 2020.

[9] AWID dataset - wireless security datasets project, 2014. Available online: http://icsdweb.aegean.gr/awid/

[10] Francisco D. Vaca and Quamar Niyaz, "An Ensemble Learning Based Wi-Fi Network Intrusion Detection System (WNIDS)," 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 2018, pp. 1-5.

[11] Aminanto, M.E., Kim, K. (2017). Detecting Impersonation Attack in WiFi Networks Using Deep Learning Approach. In: Choi, D., Guilley, S. (eds) Information Security Applications. WISA 2016. Lecture Notes in Computer Science(), vol 10144. Springer, Cham.