Name2Cat: A Lightweight Autonomous Systems Classifier Using Organization Names

Martin Thodi
University of Cape Town
South Africa
thdmar002@myuct.ac.za

Josiah Chavula
University of Cape Town
South Africa
josiah.chavula@uct.ac.za

Amreesh Phokeer
Internet Society
Mauritius
phokeer@isoc.org

***Abstract*—The Internet's backbone consists of Autonomous Systems (ASes), each typically managed by a single organisation and providing a related set of services. Accurate AS classification is crucial for understanding various aspects of Internet infrastructure, including network performance and the economic behaviours of the organisations that manage them. This granular insight is invaluable for network operators and researchers alike. In this paper, we propose a lightweight method for classifying ASes based solely on the names of the ASes and their owning organisations. By employing text feature extraction techniques, we convert these names into numerical features suitable for machine learning models. Our approach achieves an overall accuracy of 80%, with F1-scores ranging from 70% to 92% across six different categories. The method performs particularly well in categories with distinctive naming conventions, which aid classification while facing challenges in categories like Transit that have less distinctive naming patterns. Although our approach uses fewer categories than the 17 found in the state-of-the-art ASdb system, which relies on a mix of public and proprietary datasets to achieve accuracies between 75% and 93%, it offers a quick and resource-efficient solution for AS classification when detailed AS information is unavailable.**

***Index Terms*—autonomous systems, classification, Internet, Ridge classification, TF-IDF, machine learning**

I. Introduction

The Internet is composed of Autonomous Systems (ASes), each typically owned by a single organization and providing a related set of services. ASes offer valuable granularity for understanding various aspects of the Internet such as hosts, services, and economics, which is crucial for network operators and researchers. One key aspect of an AS that interests these stakeholders is its category. Knowing the category of an AS provides insights into the inner workings of the Internet, and it helps understand various interconnection phenomena such as network performance, communication infrastructure stability and reliability, and the economic incentives and behaviours of ASes. Consequently, AS classification is an active area of research.

The current state-of-the-art AS classification system is ASdb by Ziv et al. [1], which achieves accuracy between 75% and 93%. However, it relies on a mix of public and proprietary (paid) datasets to achieve this performance. In our research, we propose a lightweight machine learning-based AS classification approach that uses only the names of an AS and the organization that owns it to predict its category. By training on a publicly available dataset, we achieve good predictive performance while providing a quick and resource-efficient solution for AS classification when no additional AS information is available.

The paper is structured as follows: Section II provides the background information. Section III reviews related work. Our methodology is detailed in Section IV, followed by the results in Section V. Section VI presents the discussion, and Section VII offers the conclusion.

II. Background

In this section, we provide some background on autonomous system categories as well as textual feature extraction techniques used in this paper.

A. AS Types

ASes on the Internet fall into several categories. The categorization of ASes can be ad-hoc depending on the purpose [2]. However, three categories are widely recognized. These are *access*, *content* and *transit* providers. We adopt these categories together with three more - *enterprise*, *education/research* and *network services*. We describe these categories subsequently.

Access: Access providers are networks that sell internet services to end users [3]. They are all also known as 'eye-ball' networks, and form the majority of ASes on the internet. They include cable companies, telephone companies and wireless providers.

Content: Content Providers operate an Internet-based service but do not sell transit [3]. They typically serve as sources of traffic on the Internet, hence their traffic volumes are heavy outbound. For our purposes, this category includes cloud providers, hosting services, content delivery networks and streaming services.

Transit: An Internet Transit Provider is an autonomous system that sells access to the global Internet [3]. A transit provider serves as a bridge between the customer and the rest of the internet by announcing internet routes to the customer and vice versa.

Education/Research: Education and Research networks are specialised internet and communication service providers that operate a private network built at the national or regional level to serve and connect research and scientific communities.

Enterprise: An enterprise network is a specialized private network that connects large organizations across different locations. This category includes such institutions as banks and large retail providers.

Network Services: We use this category to classify all ASes that provide specialised networking-related services on the Internet. These include IXP route servers, BGP route collectors, and DNS root servers.

B. Text Feature Extraction

Raw text consists of symbols that cannot be directly fed into machine learning models. Therefore, to prepare text for tasks such as text classification, it must be converted into numerical representations. This process involves breaking down strings into tokens and assigning integer identifiers to each possible token, a step known as tokenization, followed by extracting features that may be useful for learning tasks. There are several approaches to text feature extraction.

One such approach is the Bag of Words (BoW). In the simplest count-based BoW method, a text corpus is represented as an unordered collection of words, disregarding grammar and word order but preserving word frequency. Each unique word in the text is treated as a feature, and its frequency of occurrence within a document is counted. This results in a matrix where rows correspond to documents and columns correspond to words, with each cell containing the word's frequency in the document. While this count-based approach is fast, it can overemphasize frequent words, such as 'the', which are not very informative for most machine learning tasks.

Another method under the BoW approach is Term Frequency-Inverse Document Frequency (TF-IDF) [4], a statistical method that balances the frequency of terms within a document with their rarity across the corpus. The TF-IDF score is calculated by multiplying two metrics: term frequency (TF), which measures how often a word appears in a document, and inverse document frequency (IDF), which assesses how common or rare a word is across all documents in the corpus. This method highlights words that are more meaningful for the specific context of each document while downplaying those that are common but less informative.

Formally, let $D = \{d_1, d_2, ..., d_N\}$ be a collection of N documents and t be a term. The TF-IDF score for term t in document d is defined as:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \cdot \text{IDF}(t, D) \tag{1}$$

where $\text{TF}(t, d)$ is the term frequency of t in d, and $\text{IDF}(t, D)$ is the inverse document frequency of t in D.

Term Frequency (TF) is typically calculated as:

$$\text{TF}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{2}$$

where $f_{t,d}$ is the raw frequency of term t in document d, and the denominator is the sum of raw frequencies of all terms in d.

Inverse Document Frequency (IDF) is usually defined as:

$$\text{IDF}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|} \tag{3}$$

where N is the total number of documents in the corpus, and $|\{d \in D : t \in d\}|$ is the number of documents where the term t appears.

The resulting TF-IDF score has several important properties. First, it increases proportionally to the number of times a word appears in the document. This aspect captures the relevance of the term within the specific document. Secondly, the score is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general. This feature ensures that common words, which may not carry significant meaning, are not given undue importance. Lastly, TF-IDF assigns higher weights to terms that are rare in the corpus but frequent in a particular document, potentially indicating that these terms are more characteristic or important for that document. This property allows the method to identify terms that are uniquely significant to specific documents.

In practice, variations of these formulas may be used to address specific characteristics of the data or to improve performance. For instance, logarithmic scaling might be applied to TF to dampen the effect of high-frequency terms, reducing the impact of words that appear very frequently in a single document. Alternatively, a smoothing term might be added to IDF to handle terms that appear in all documents, preventing division by zero and allowing the method to work with corpus-wide terms. These modifications allow the TF-IDF method to be adapted to various types of text data and different analytical needs. In this paper, we use the CountVectorizer and TfidfVectorizer implementations from Sckit-learn [5] for count-based and TF-IDF feature extraction methods, respectively.

III. Related Work

There have been several research efforts to classify Autonomous Systems and the organizations that own them. One of the earliest significant works was by Dimitropolous et al. [6], who used text classification on AS WHOIS data to categorize ASes into six categories: large ISP, small ISP, Internet eXchange Point (IXP), customer, university, and network information centres. They reported 95% coverage and 78% accuracy. Building on Dimitropolous et al.'s methodology, CAIDA maintained a publicly accessible dataset that classified ASes into transit/access, enterprise, and content categories [7]. However, due to declining accuracy, CAIDA discontinued this dataset in 2021.

Another body of research has focused on using internet topology features to infer AS types and their relationships [8], [9], [10], [11], [12]. Notably, Dhamdhere and Dovrolis [2] achieved an accuracy of 76-82% in classifying ASes into five categories: enterprise customers, content providers, small and large transit providers, and access/hosting providers.

Baumann and Fabian [13] introduced another methodology, classifying ASes into 10 categories: communication, construction, consulting, education, entertainment, finance, healthcare, transport, travel, and utilities. Their approach achieved 57% coverage. Ziv et al. [1] proposed a state-of-the-art methodology for classifying ASes by addressing inconsistencies in data collection and classification from Regional Internet Registries. Their system, ASdb, uses data from business intelligence databases and machine learning to achieve 96% coverage of ASes, with 93% accuracy on 17 industry categories and 75% accuracy on 95 sub-categories. There are also commercial datasets that offer proprietary AS classification. A well-known example is provided by IPinfo [14], which classifies ASes into ISP, hosting, education, and business.

This work proposes a low-resource approach to AS classification. By using only AS and organisation names, we achieve good predictive performance. Although our approach uses fewer categories than the 17 found in the state-of-the-art ASdb, it offers a quick and resource-efficient solution for AS classification when no additional information about the AS is available.

IV. Methodology

A. Data Description

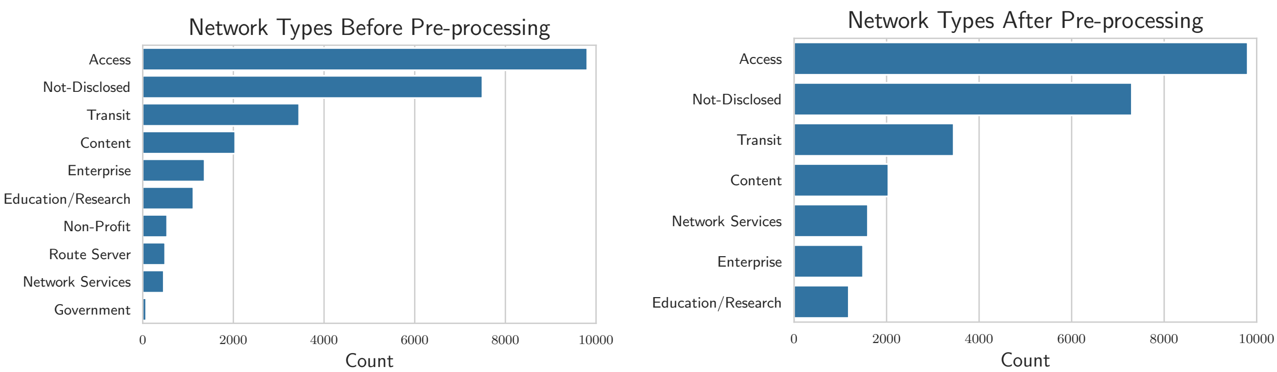
An AS must establish connections with other ASes to achieve global reachability [3]. These relations can be either transit (customer-provider) or peering relationships. In a transit relationship, the customer AS pays the provider AS to route its traffic to the rest of the Internet. Conversely, in a peering relationship, two ASes agree to exchange traffic (including that of their immediate customers) without any payment. ASes seeking peering opportunities often list their profiles, requirements, points of presence, and contact information in PeeringDB [15], a free and open-source database. CAIDA maintains publicly available daily snapshots of the PeeringDB database [16]. We downloaded and used the snapshot from May 31, 2024. PeeringDB organizes its data into several tables, but we focused on two specific tables: *net* and *org*. The *net* table contains information about an AS, while the *org* table provides details about the organisation that owns the AS. A single organisation can own multiple ASes. From the *org* table, we used only the organisation's name, and from the *net* table, we extracted the name of the AS.

B. Data Preprocessing

PeeringDB [15] provides eleven possible non-exclusive categories for the AS type field; NSP, Content, Cable/DSL/ISP, Enterprise, Educational/Research, Non-Profit, Route Server, Network Services, Route Collector, Government and Not-Disclosed. Network operators assign these categories to their ASes when they register their ASes in peeringDB. We renamed the NSP category Transit and the Cable/DSL/ISP to Access. To make the classes exclusive; we combined Route Server, Route Collector and Network Services into a single category called Network Services, while the Government and Non-Profit categories are collapsed into Enterprise. Thus, in the end, we have the six categories - *Transit, Content, Access, Enterprise, Education/Research* and *Network Services*. Figure 1 shows the distribution of ASes by type before and after pre-processing.

We also noted that there were many ASes whose type was Not-Disclosed. To enhance our datasets, we applied rules to label some non-disclosed ASes where it was clear to do so. Records containing the terms *'Bank', 'Retail', 'Investment', 'Financial'*, or *'Banco'* were categorized as Enterprise, resulting in 44 records being classified this way. Those containing *'University', 'College'*, or *'Institute of Technology'* were categorized under Education/Research, with 61 records falling into this category. Records that included *'Ministry'* were categorized as Enterprise (from Government), accounting for 4 records. Finally, records containing 'F-ROOT' or 'Route Server' were classified as Network Services, with 82 records categorized accordingly.

Finally, we created a combined input name by merging the AS name with the organization name in the following format: *AS_Name–Organization_Name*. For example, Akamai (AS23454), owned by Akamai Technologies Inc., is combined to form *Akamai–Akamai Technologies Inc*. This combined name is then used as input to the tokenizer.



(a) Distribution of ASes before pre-processing

(b) Distribution of ASes after pre-processing.

Fig. 1: Distribution of ASes

C. Pipeline

To achieve our text classification task, we use the following pipeline;

1) *Tokenization*: The raw text is split into smaller units, such as words or subwords, known as tokens.
2) *Feature Extraction*: The tokens are transformed into numerical representations (features) that capture the essential characteristics of the text.
3) *Classification*: A machine learning model uses these features to categorize the text into predefined classes. We employed RidgeClassifier [5] for our use case.
4) *Output*: The final classification results are produced, indicating the category of the AS.

D. Training and Test Data

The dataset was divided into training and testing sets to ensure a comprehensive evaluation of our model. We allocated 87% of the data for training purposes while reserving the remaining 13% for testing. This distribution allows for a robust training process while still maintaining a sufficient sample for evaluating the model's performance on unseen data.

To address the challenge of class imbalance in our dataset, we implemented both oversampling and downsampling techniques. Initially, we identified two minority classes: Enterprise, which originally contained 1,492 samples, and Education/Research, which had 1,184 samples. To balance these underrepresented classes, we employed an oversampling method to increase their sizes to 1,500 samples each. This process involved using SMOTE [17] to create synthetic samples or duplicate existing ones to reach the target number.

Simultaneously, we addressed the majority classes in our dataset. To prevent these larger classes from dominating the model's learning process, we applied a downsampling technique. Each majority class was reduced to 1,500 samples, matching the size of the oversampled minority classes. This balanced approach ensures that all classes have equal representation in the training data, with 1,500 samples each.

By implementing these sampling techniques, we aimed to create a more balanced dataset that would allow the model to learn equally from all classes, potentially improving its overall performance and reducing bias towards any particular class.

E. Model Configuration

We used two text vectorisation techniques: CountVectorizer and TfidfVectorizer. Both vectorizers were configured with an n-gram range of (1,3), allowing them to capture individual words and phrases up to three words long.

For classification, we used a Ridge Classifier with the 'auto' solver and an alpha value of 1. This configuration allows the model to automatically select the most appropriate solution method while maintaining a balance between model complexity and generalisation.

All computations were performed on a machine with an 11th Generation Intel Core i5 processor, featuring 8 cores at 2.6 GHz and 16 GB of RAM. On average, the training process was completed in just 187 milliseconds.

F. Evaluation Metrics

We evaluated the model using *precision, recall, F1 score* and *overall accuracy* metrics. Precision is the fraction of true positives among all positive predictions. It is used to measure the model's accuracy in identifying positive instances of a class. Recall measures a model's sensitivity to capturing all positive instances of a class. It is computed as a fraction of a class's true positive among all actual positive instances. F1-score is a harmonic mean of precision and recall.

V. Empirical Results

In this section, evaluate the models using precision, recall, F1-score and accuracy metrics.

Table I presents the evaluation metrics (precision, recall, and F1-score) for two classification models: one using CountVectorizer and the other using TF-IDF Vectorizer. The TF-IDF model generally outperforms the CountVectorizer model across most categories. This is expected, as the TF-IDF feature extraction method was introduced to address the weakness of count-based approaches.

The TF-IDF model shows notable improvements in the *Access* and *Transit* categories, with precision, recall, and F1-scores consistently higher than the CountVectorizer model. This suggests that the TF-IDF approach is more effective at capturing the distinctive features of these categories.

In the *Education/Research* category, we observe an interesting trade-off. The CountVectorizer model achieves higher precision (0.95 vs 0.85), while the TF-IDF model excels in recall (0.96 vs 0.91). This indicates that the CountVectorizer is more conservative in its predictions for this category, resulting in fewer false positives, while the TF-IDF model identifies a greater proportion of the actual *Education/Research* instances.

TABLE I: Evaluation Metrics

	Precision		Recall		F1-Score		Support
	CountVect	TF-IDF	CountVect	TF-IDF	CountVect	TF-IDF	
Access	0.62	0.70	0.65	0.70	0.63	0.70	192
Content	0.73	0.76	0.75	0.76	0.74	0.76	216
Education/Research	0.95	0.85	0.91	0.96	0.93	0.90	206
Enterprise	0.85	0.85	0.81	0.82	0.83	0.84	200
Network Services	0.89	0.93	0.88	0.91	0.89	0.92	181
Transit	0.67	0.73	0.68	0.66	0.67	0.70	205
Macro avg	0.78	0.80	0.78	0.80	0.78	0.80	1200
Weighted avg	0.78	0.80	0.78	0.80	0.78	0.80	1200

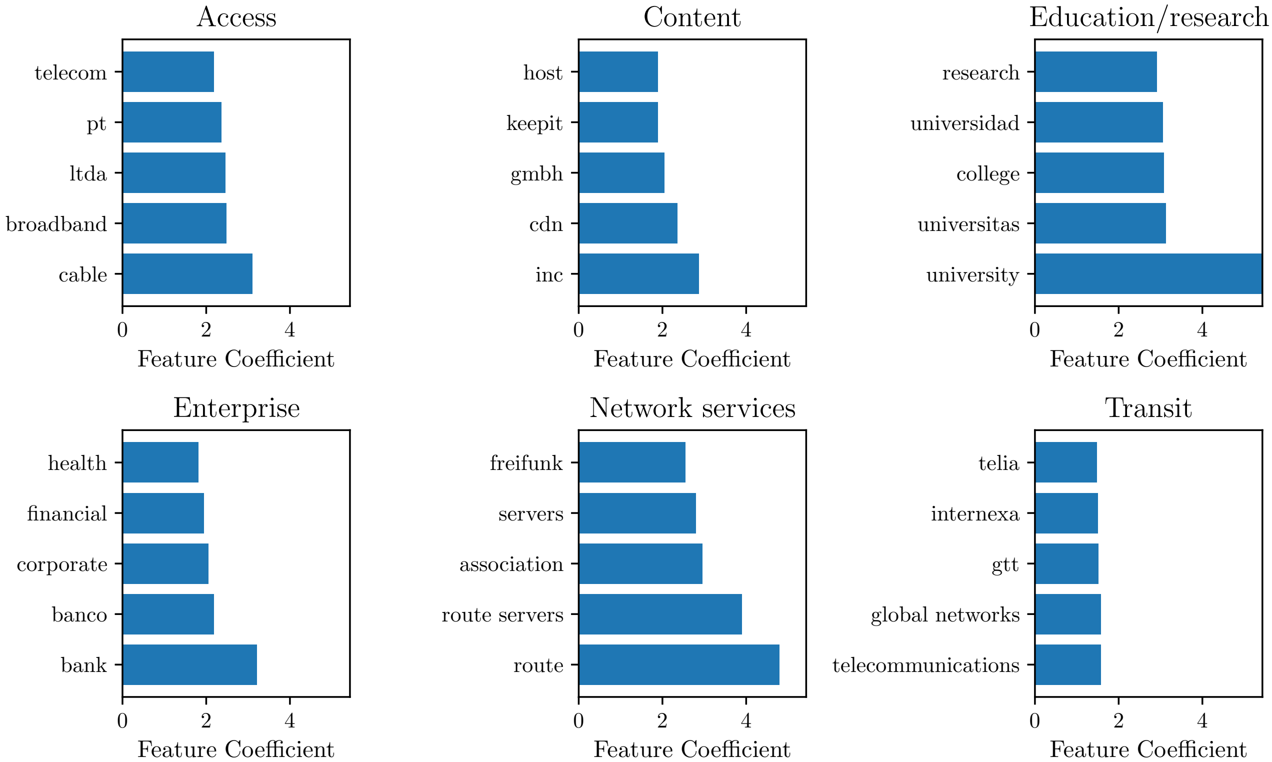


Fig. 2: Most predictive words per category

For the *Enterprise* and *Network services* categories, both models perform similarly, with only slight variations in their metrics. This suggests that these categories may have distinct naming conventions that are well-captured by both approaches.

The *Content* category sees a modest improvement across all metrics with the TF-IDF model, indicating that this approach better represents the nuances of content-related text.

While both models demonstrate strong performance, the TF-IDF model appears to have a slight advantage overall, particularly noticeable in categories that may have been more challenging for the count- based approach, such as Access and Transit.

A. *Feature Importance Analysis*

The RidgeClassifier we employed is a linear model where the target variable is modelled as a linear combination of the input features. The coefficients of a linear model indicate how the output changes when a given feature is varied while keeping all other features constant. In this section, we briefly analyze the most important features by examining the coefficients of the model trained using TF-IDF feature extraction.

The Education/Research category stands out with highly predictive words, especially "university," which has a coefficient near 4, indicating clear distinctions for this category. The Network Services category also features distinctive and predictive words such as "route servers," suggesting a consistent naming of ASes and their organizations in this category. An interesting predictive word here is "freifunk," which is German for free-radio and corresponds to non-commercial wireless community network ASes in Germany [18] listed as Non-Profit in PeeringDB—a category we collapsed into Network Services. The Enterprise category has strong predictor words like "bank" and "electric," indicating consistent and informative naming conventions used by the organizations owning these ASes. While the Content category has somewhat predictive words like "inc," they are not distinct enough for the category, resulting in low precision and confusion with Access and Transit classes. Transit has the least predictive and distinct words, leading to confusion with the Access class and resulting in the lowest metrics overall. We believe that including more features would improve both precision and recall for these classes, as demonstrated by approaches like ASdb that utilize more features.

VI. Discussion

The primary goal of this research was to develop a lightweight method for classifying ASes using only the names of the ASes and their owning organisations. We utilised count-based and TF-IDF text feature extraction techniques to convert these names into numerical features suitable for machine learning models, specifically employing a RidgeClassifier. The method achieved an overall accuracy of 68%, with F1 scores ranging from 55% to 82% across six categories.

A. Importance of the Findings

The most significant finding of our study is the ability to classify ASes with good accuracy using a lightweight and low-resource method. This is particularly important because it offers a practical solution for scenarios where detailed AS information is unavailable. The method's performance is notably strong in categories with distinctive naming conventions, demonstrating the potential for effective classification based on textual data alone. This approach provides a valuable tool for network operators and researchers, enabling them to gain insights into network performance, communication stability, and economic behaviours of ASes with minimal resource investment.

B. Study Limitations

The generalisability of our results may be limited by the specific dataset employed and the inherent biases in AS naming conventions. For example, the study predominantly used names based on the English language. As a result, its performance may diminish when classifying ASes with names in languages markedly different from English. Additionally, our method encountered difficulties with categories that have less distinctive naming patterns, which could impact overall classification accuracy. Nonetheless, the proposed approach has shown promise in using names alone to classify autonomous systems into various categories.

C. Suggestions for Future Work

Future research should investigate the trade-off between the accuracy gained by incorporating additional features, such as AS topology, traffic patterns, and other network-related data, and the model's simplicity and data efficiency. This balance is crucial for improving classification accuracy, particularly in challenging categories like transit and access. Moreover, future studies could consider introducing more categories that reflect the economic sectors ASes belong to, such as Real Estate, Construction, and similar industries, as is done in state-of-the-art classification systems like ASdb.

In our ongoing work, we are using the findings from this study to develop an optimal peer selection framework. This framework leverages the classified AS categories to create realistic internet models that take into account the preferences and requirements of different AS types.

VII. Conclusion

In this paper, we proposed a lightweight approach to classify Autonomous Systems using only the names of ASes and their owning organizations. By leveraging text feature extraction techniques like CountVectorizer and TF-IDF Vectorizer, we transformed these names into numerical features for machine learning models, specifically employing a RidgeClassifier for prediction. The proposed method achieved an overall accuracy of 80%, with F1-scores ranging from 70% to 92% across different categories. The model performed best in the Education/Research and Network Services categories, indicating that certain AS categories have more distinctive and predictive naming conventions. However, categories like Transit showed lower predictive power, resulting in more classification errors.

Despite its merits, our approach has limitations. The reliance on names alone may miss important information needed for higher accuracy. Adding features such as AS customer cones, traffic patterns, or more detailed organizational data could improve performance. While down-sampling addressed class imbalance, it may have led to information loss from larger classes. Compared to the state-of-the-art ASdb system, which achieves higher accuracy, the proposed method's performance highlights the trade-off of using simpler features. Nevertheless, it offers a quick and resource-efficient alternative, useful when additional AS information is not available. Future work could focus on integrating more features and advanced techniques to enhance classification performance.

References

[1] Maya Ziv, Liz Izhikevich, Kimberly Ruth, Katherine Izhikevich, and Zakir Durumeric. Asdb: a system for classifying owners of autonomous systems. In *Proceedings of the 21st ACM Internet Measurement Conference*, IMC '21, page 703–719, New York, NY, USA, 2021. Association for Computing Machinery.

[2] Amogh Dhamdhere and Constantinos Dovrolis. Ten years in the evolution of the internet ecosystem. In *ACM/SIGCOMM Internet Measurement Conference*, 2008.

[3] William Norton. *The Internet Peering Playbook: Connecting to the Core of the Internet*. DrPeering Press, 2014 edition edition, 2014.

[4] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[6] Xenofontas A. Dimitropoulos, Dmitri V. Krioukov, George F. Riley, and Kimberly C. Claffy. Revealing the autonomous system taxonomy: The machine learning approach. *ArXiv*, abs/cs/0604015, 2006.

[7] CAIDA. As classification. https://www.caida.org/catalog/datasets/as-classification/. (Accessed on 05/13/2024).

[8] Enze Liu, Gautam Akiwate, Mattijs Jonker, Ariana Mirian, Stefan Savage, and Geoffrey M. Voelker. Who's got your mail?: characterizing mail service provider usage. *Proceedings of the 21st ACM Internet Measurement Conference*, 2021.

[9] Matthew Roughan, Walter Willinger, Olaf Manuel Maennel, Debbie Perouli, and Randy Bush. 10 lessons from 10 years of measuring and modeling the internet's autonomous systems. *IEEE Journal on Selected Areas in Communications*, 29:1810–1821, 2011.

[10] Reza Motamedi, Reza Rejaie, and Walter Willinger. A survey of techniques for internet topology discovery. *IEEE Communications Surveys & Tutorials*, 17:1044–1065, 2015.

[11] Vasileios Giotsas, Christoph Dietzel, Georgios Smaragdakis, Anja Feldmann, Arthur W. Berger, and Emile Aben. Detecting peering infrastructure outages in the wild. *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 2017.

[12] Srinivas Shakkottai, Marina Fomenkov, Ryan Koga, Dmitri V. Krioukov, and Kimberly C. Claffy. Evolution of the internet as-level ecosystem. *The European Physical Journal B*, 74:271–278, 2006.

[13] Annika Baumann and Benjamin Fabian. Who runs the internet? - classifying autonomous systems into industries. In *International Conference on Web Information Systems and Technologies*, 2014.

[14] IPinfo. The trusted source for ip address data, leading ip data provider - ipinfo.io. https://ipinfo.io/. (Accessed on 05/13/2024).

[15] PeeringDB. Peeringdb. https://peeringdb.com/. (Accessed on 05/13/2024).

[16] CAIDA. Peeringdb archive. https://www.caida.org/catalog/datasets/peeringdb/. (Accessed on 05/13/2024).
[17] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.
[18] Freifunk.net. What is freifunk? https://freifunk.net/en/what-is-it-about/. (Accessed on 05/13/2024).