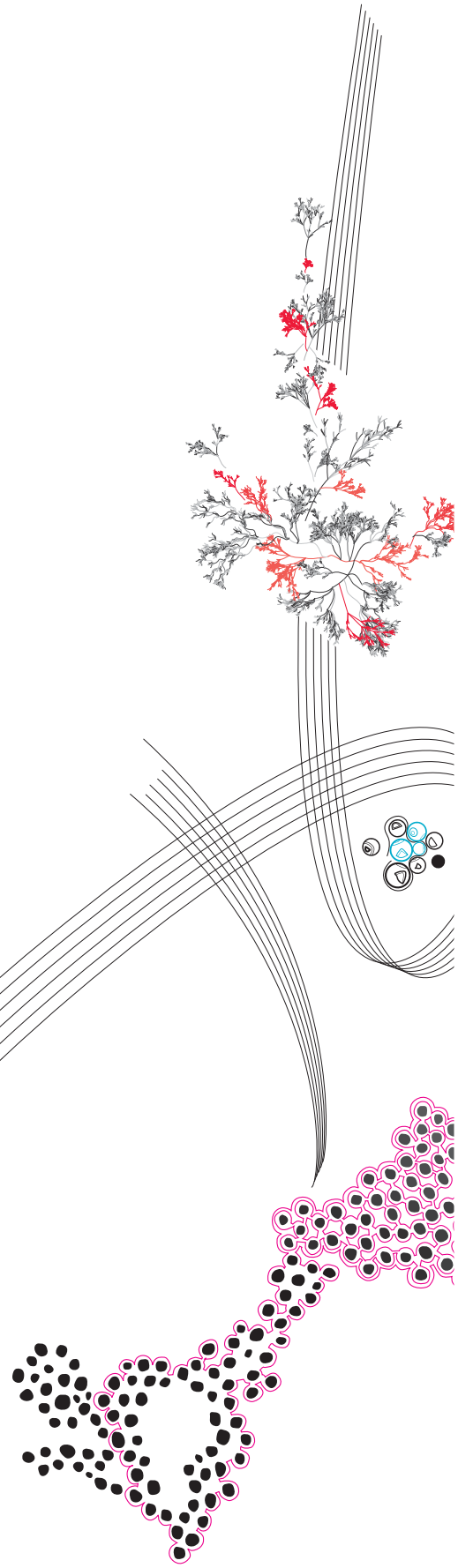

MSc Computer Science
Final Project

Tracking Internet Host Usage Types: Classification and Trends via Reverse DNS Data

Roman Khavrona

Committee:
Mattijs Jonker
Thijs van Ede
Roland van Rijswijk-Deij

September, 2024

Department of Computer Science
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

Contents

Abstract

The rapid growth of the Internet has led to increased complexity in IP address usage, making it challenging to effectively track and understand the functions these IP addresses serve, such as whether they belong to datacenters, educational institutions, internet service providers or other categories. This topic has not been widely researched publicly and is crucial for network management, cybersecurity and content regulation. Given that IP addresses are merely numerical labels, we utilized their associated string hostnames obtained via reverse DNS for our classification tasks. We analyzed 1,285,834,541 IPv4 addresses (approximately 30% of the total IPv4 address space) and classified each IP address by its usage type using IP2Location's commercial dataset as the ground truth. Datasets covering these types of data are expensive to obtain and have unknown methodologies; therefore, we aimed to create a model with an open methodology that would output the usage type by only providing a hostname, which is easily obtainable by anyone performing a reverse DNS query with the IP address of interest.

In our research, we analyzed both manually crafted and automatic features generated through the Word2Vec technique from hostnames and supplied these features to machine learning models, achieving a prediction accuracy close to 70%. Additionally, we performed a longitudinal analysis of usage types throughout 2023, utilizing four different data snapshots to identify trends and tendencies in shifts of IP usage types, and observed notable changes, such as the consistent decline in the organization (ORG) category, overall decrease in mobile ISP services, the steady increase in datacenter IP addresses and expansion in educational IP address usage. Lastly, we investigated the potential of applying our developed techniques to predict the country attribute within the IP2Location dataset. This attribute was selected due to IP2Location's claim of high accuracy, a claim that has been substantiated by other researchers who have reported accuracy levels approaching 100%. Our objective was to assess the performance of our techniques on an attribute with established accuracy, and the results demonstrated an accuracy exceeding 90%, potentially indicating that our methodology for inferring usage types is practical for real-world scenarios.

Keywords: IP address usage type, reverse DNS, hostnames, Word2Vec, machine learning, longitudinal analysis

Chapter 1

Introduction

The rapid growth and increasing complexity of the Internet over recent decades, which has become a central aspect of our personal and professional lives, have made it a subject of significant interest in research circles. The Internet, which began as a basic network linking devices in a small area, has transformed into a vast system connecting millions of devices globally. This transformation has captured the attention of researchers, leading them to explore the Internet's topology [87, 39, 38], security [48, 27], user experience [32] and other components.

Following the exploration of the Internet's expansive and complex network, another area of growing interest emerges: delineating Internet hosts by categories that reflect their primary functions, such as commercial, datacenters, content delivery networks (CDNs), or others. Such interests are not merely academic; they are of practical importance to governments and large corporations, particularly in the realms of cybersecurity and content regulation.

One notable instance of corporate interest in this area is seen in the actions of companies like Netflix, which are actively working on identifying IP addresses associated with VPN services to subsequently block access through those VPNs [21, 12]. The motivation behind these efforts is multifaceted, often tied to legal and operational constraints. For example, Netflix's licensing agreements prohibit the broadcasting of certain content in specific countries, and the use of VPNs can bypass these geographic limitations, leading to potential violations of copyright agreements. This scenario highlights the significant implications of identifying the nature of Internet traffic and the services associated with particular IP addresses. Moreover, one of our main motivations for this research is that most studies on the primary functions of IP addresses, particularly those involving large-scale data analysis, are conducted by major commercial companies using proprietary methodologies that are not publicly available or peer-reviewed. Therefore, it is of great interest to explore this topic within the academic community.

In our research, we broaden our scope beyond merely identifying VPN service IP addresses (which potentially fall into datacenter category as we show in Chapter 5), aiming to categorize a wider range of IP addresses and uncover the diverse types of content they represent. While IP addresses are essentially numerical labels separated by dots, they lack inherent contextual meaning. This highlights the need for analyzing hostnames, which typically include textual information and are directly connected to their corresponding IP addresses. To acquire these hostnames, we utilize daily hostname lookup data [93], capturing a comprehensive view of a big segment of the IPv4 address space. This method allows us to associate each IP address with a hostname, enhancing our ability to analyze and categorize the vast array of content present on the Internet.

Acquiring high-quality data for determining the primary functions (usage types) of IP addresses is a challenging task. This difficulty arises not only because methodologies are often unpublished or not peer-reviewed, leading to unproven quality, but also because such data is rare and expensive to obtain. Nevertheless, to establish a reference point for our research, to train and assess our models, we utilize the IP information dataset provided by IP2Location [14]. We select IP2Location over other popular providers such as MaxMind [16], IPinfo [10] and others due to its inclusion of a usage type column, which is of primary interest to us. To verify that the data is usable as ground truth for the evaluation and training of the models, we perform some quality checks (see Chapter 5).

The remainder of the paper is organized as follows: Chapter 2 presents the main goals of the research and research questions we aim to address. In Chapter 3, we provide an overview of key background topics essential for understanding different parts of our research. Chapter 4 covers the related work in the field of hostname analysis. In Chapter 5, we discuss data labeling, preprocessing and validation. In Chapters 6, 7, and 8, we outline our methodology and present the experiments and findings that address our research questions. Finally, we discuss the limitations of our study and suggest potential directions for future research.

Chapter 2

Main goals and research questions

As discussed in Chapter 1, the motivation for our research arises from the lack of public research on IP address usage. Commercial companies often withhold their work and solutions from the academic community. Although IP2Location produces relevant data, its creators do not disclose their methodology for inferring usage types. This makes it particularly interesting to explore how predictions based on hostnames can address this task, especially since hostnames can be easily obtained for IP addresses through reverse DNS queries (refer to Chapter 3 for background information on DNS).

To achieve our objectives, we integrate supervised methods by leveraging our reference IP2Location data (or we can say *ground truth* assuming IP2Location data is highly accurate). Moreover, we explore the possibility of applying unsupervised and zero-shot learning methods that do not rely on ground truth during the training phase but use the usage type data as a reference point during assessment. We explore these methods because obtaining large amounts of high-quality labeled usage type datasets is challenging. While we perform some validation of the usage type data (see Chapter 5), we cannot definitively verify its quality due to large dataset's size. Therefore, it is interesting to investigate whether the model training phase can be conducted without relying solely on ground truth, while still validating the models using the available IP2Location dataset.

Receiving daily reverse DNS data and IP2Location data throughout 2023 allows us to comprehensively analyze IP-hostname data over time. Our specific focus is on shifts in usage types and assessing whether these shifts can be detected solely through hostname data. This information can be useful for enhancing the understanding of Internet infrastructure dynamics.

Lastly, we also explore the other attribute that can be predicted based on hostnames, such as the country in which the IP address is located. This information is also available within the IP2Location dataset and is the only attribute that IP2Location claims to have very high accuracy (over 99%). Hence, it is of interest to see how our methodology for inferring usage type performs in this prediction, considering there are claims from IP2Location developers and the IP2Location accuracy of the country-level attribute was also explored and proven in previous research to be close to 100% [57].

Hence, based on aforementioned key points, we define following research questions for our research:

Research Question 1 (RQ1): How effectively can IP addresses be categorized in terms of the categories of content they represent using reverse DNS data?

By addressing this research question, we aim to find the best method for accurately predicting the usage type of IP hosts.

Research Question 2 (RQ2): Can we identify shifts and trends in the usage types of IP addresses over time?

By addressing RQ2, we aim to analyze the evolution of Internet host usage type categories to shed light on potential dynamicity within Internet hosts. Additionally, we seek to assess whether hostnames are suitable for tracking IP usage type categories over time.

Research Question 3 (RQ3): How effectively can the country attribute of IP addresses be predicted using reverse DNS data?

This question aims to explore the potential of applying techniques we develop to answer RQ1 using hostname data to predict the country of IP addresses. Additionally, it seeks to validate our methodology, as there is evidence of high accuracy in the country attribute of the IP2Location dataset.

Chapter 3

Background

In this Chapter, we provide an overview of the key background topics essential for understanding different parts of our research.

First, we discuss the Domain Name System (DNS) and reverse DNS, as they are fundamental to our research. Understanding how DNS maps domain names to IP addresses through forward DNS and how reverse DNS maps IP addresses back to hostnames is crucial for comprehending the data we obtain via reverse DNS measurements.

Following this, we discuss common preprocessing techniques for textual data and their potential application to hostnames in our research. Later in Chapter 5, we investigate the applicability of these methods and select the most suitable candidates for our analysis.

Then, we address the importance of feature engineering and feature selection for both supervised and unsupervised methods in our research. This subsection encompasses a discussion on manually engineered features derived from domain knowledge and the methods used to validate their relevance to the target variable. Additionally, we discuss automatic feature extraction from textual data using advanced techniques such as Word2Vec for generating word embeddings, which we use in our research.

Subsequently, we describe supervised and unsupervised learning, explain their differences, and outline the advantages and disadvantages of each approach. Next, we provide an overview of common supervised and unsupervised models available for implementation within the PySpark data analytics engine [18], which we use for our research (later elaborated in Chapter 6), and discuss the metrics available within the engine to evaluate these approaches.

Finally, we discuss zero-shot learning approach using the BERT zero-shot classifier for our hostname classification task. The model and the underlying concepts are explained, along with a discussion on its application and potential benefits.

By covering these topics, we aim to provide a comprehensive foundation that will enable readers to follow our research approach and understand the rationale behind the selected techniques and approaches.

3.1 Domain Name System (DNS)

3.1.1 Forward DNS

The Domain Name System (DNS) represents a fundamental aspect of internet functionality, serving as a phonebook of the Internet [92, 72, 53, 25, 37]. It facilitates the mapping of human-readable domain names entered into web browsers, email servers, databases, cloud services and networking utilities, to their corresponding Internet Protocol (IP) addresses,

a process termed `forward DNS`.

When a user inputs a domain such as *example.com* into the search bar of a browser, the DNS ensures the accurate rendering of the appropriate service or website. This mechanism is paramount given that each device connected to the Internet is assigned a unique IP address, which is challenging for users to recall. The establishment of the DNS was therefore instrumental in enabling users to utilize memorable domain names, thereby alleviating the complexity of direct IP address usage.

3.1.2 Reverse DNS (rDNS)

While the mapping from domain names to Internet addresses is crucial for Internet usage, the reverse mapping, known as `reverse DNS (rDNS)`, holds significant importance as well. Reverse DNS involves converting IP addresses, which are the identifiers of hosts, into hostnames. This facilitates various functions such as network management and troubleshooting, security checks, load balancing, and it also has applications in research, such as inferring geolocation from hostnames [64, 37, 51, 84].

For reverse DNS lookups, the owner of the network prefix is tasked with establishing reverse zones or can delegate the management to external DNS services [19, 73, 92, 24]. These reverse zones, which are distinct segments of the DNS, contain PTR (Pointer) records, crucial for the reverse DNS process. PTR records within a reverse zone serve to map IP addresses back to their corresponding hostnames. The specific zone utilized for IPv4 address mapping is the *in-addr.arpa.* zone. This particular zone plays a key role in linking IPv4 addresses with hostnames, operating in accordance with the standards of the DNS architecture. Similarly, the reverse zone for IPv6 addresses uses the *ip6.arpa.* domain, following a parallel approach but tailored for IPv6 [9].

Additionally, it is important to note that reverse DNS lookups typically yield either no hostname or a single hostname [37, 29]. This is because hosts do not necessarily have a hostname record in reverse DNS, and when they do, they usually have one PTR record, though this is not always the case. Another key point to mention is that the hostname obtained from a reverse DNS query may not always match the hostname found in a forward DNS query. Ensuring that the forward DNS and reverse DNS entries match is known as Forward Confirmed reverse DNS (outlined in [29]). This consistency is crucial for applications such as reducing email spam by verifying the authenticity of email sources.

3.2 Data preprocessing

In our research, we work with a dataset that includes hostnames, which we aim to analyze and categorize based on the type of content these hostnames represent. A crucial initial step in handling text data is the preprocessing phase, as it ensures that the fundamental units of text are accurately prepared and passed to further processing stages [94, 54, 59, 50], such as, for instance, to actual prediction models. Below, we list and discuss common preprocessing techniques explored in our research, and later in Chapter 5, we discuss our choices in selecting or not selecting them for our study.

Noise removal

Removing noise is a vital step in text analysis [59], and this could potentially apply to our work with hostnames as well. In text data preprocessing, noise removal often involves eliminating special characters or punctuation. For hostnames, this process could involve

ignoring delimiters or omitting parts that we may find irrelevant to our analysis upon further research.

Tokenization

In our analysis of hostnames, we investigate tokenization. Tokenization, a standard process in text analysis, involves breaking down a string of text into smaller units, such as words or abbreviations [54, 50, 59].

In the DNS, there is a hierarchy of names defined, starting with the root, which is unnamed and often symbolized by a dot [1, 78]. This is followed by the Top-Level Domain (TLD), separated by a dot, then by the Second-Level Domain (SLD) and potentially by further subdomains [3, 1, 78]. For instance, in `example.com`, `com` is the TLD and `example` is the SLD. This structure suggests that hostnames can be split (tokenized) using the dot as a delimiter, separating the different parts for further processing.

Capitalization

When processing text, handling capitalization is crucial [50, 59]. In the case of hostnames, this means lowercasing all letters to ensure uniformity. Hostnames are case-insensitive as per RFC 4343 [40], meaning that different capitalization styles, such as *Example.com* and *example.COM*, refer to the same hostname. This suggests that standardizing capitalization by converting all letters to lowercase may be safe and can simplify further processing by ensuring greater uniformity.

Abbreviation

Another important step during the preprocessing stage is addressing the presence of abbreviations or shortened versions of words [59, 50]. These variations in text need to be identified and handled appropriately to ensure accurate and consistent analysis.

Stemming

We also investigate stemming, which involves reducing words to their base or root form [94, 54, 59, 50]. This process is useful because words can appear in various forms while retaining the same semantic meaning. For example, stemming would reduce variations like *connecting*, *connected*, and *connection* to the root form *connect*.

Lemmatization

Lemmatization could be used for the words within hostnames, potentially as an alternative to stemming. This technique aims to strip away the suffixes of words, bringing them down to their basic form [94, 59, 50]. For instance, if hostnames include verbs with different tenses like *runs.example.com*, *running.example.com*, and *ran.example.com*, lemmatization would convert all variations to their lemma form *run.example.com*.

3.3 Manual feature engineering

Feature engineering is a critical phase in supervised and unsupervised learning, as it involves identifying the most informative attributes of the data that contribute to the performance of predictive and clustering models [75, 49].

Manual feature engineering involves domain expertise and a deep understanding of the specific case of interest [75, 69]. This approach is particularly practical in situations where certain features are known to have significant importance based on prior knowledge or related work. For instance, for our research, there is a pre-existing notion that hostname length is an important feature due to the nature of certain services, such as CDNs, which often have long hostnames. In our study, we discuss manual feature engineering for hostname-like data within the context of previous research in the Related Work chapter (see Section 4.2). We highlight how expert knowledge can guide the selection of features to achieve more accurate and meaningful analysis.

3.4 Manual feature relevance

In the context of our research, determining feature relevance of crafted manual features is crucial for enhancing the performance of both supervised and unsupervised learning models. Feature relevance helps identify which attributes of the data contribute most significantly to the predictive power and accuracy of the models [60, 56]. Since our features are numerical (described later in Section 6.1) and the target variable is categorical, we employ the ANOVA F-test to assess feature relevance. Using the results of this test, we can determine which features are likely to enhance model accuracy.

3.4.1 ANOVA F-test

The Analysis of Variance (ANOVA) F-test is a statistical method used to determine if there are significant differences between the means of two or more groups [90, 55, 13, 88, 41]. It is commonly used in feature selection to assess the importance of numerical features with respect to a categorical target variable. The null hypothesis (H_0) for the ANOVA test states that all group means are equal, while the alternative hypothesis (H_A) states that at least one group mean is different.

Mathematically, the F-statistic is calculated as the ratio of the between-group variance (MSB) to the within-group variance (MSW):

$$F = \frac{\text{MSB}}{\text{MSW}}$$

The between-group variance, MSB, measures the variability of the group means around the overall mean and is given by:

$$\text{MSB} = \frac{\sum_{i=1}^{k} n_i (\bar{X}_i - \bar{X})^2}{k - 1}$$

where k is the number of groups, n_i is the sample size of group i, $\bar{X}_i$ is the mean of group i, and $\bar{X}$ is the overall mean.

The within-group variance, MSW, measures the variability within each group and is calculated as:

$$\text{MSW} = \frac{\sum_{i=1}^{k} \sum_{j=1}^{n_i} (X_{ij} - \bar{X}_i)^2}{N - k}$$

where N is the total sample size and X_{ij} is the value of the j-th observation in group i.

In feature selection, the p-value corresponding to the F-statistic indicates the probability of observing such a value under the null hypothesis. If the p-value is less than a specified threshold (e.g., 0.05), the null hypothesis is rejected, suggesting that the feature significantly contributes to differentiating between groups.

3.5 Automatic feature extraction

Automatic feature extraction, on the other hand, leverages algorithmic approaches to identify relevant features from large datasets [61, 44]. Techniques such as Word2Vec, a popular word embedding method, is employed for this purpose in our research.

3.5.1 Word2Vec

Word2Vec is a widely used word embedding technique that converts text data into continuous vector representations [82, 71, 70, 35, 67, 26, 7, 100]. Developed and supported by Google, Word2Vec captures semantic relationships between words, which enhances the ability of models to understand and process textual information more effectively. This technique employs two primary learning models: Continuous Bag of Words (CBOW) and Skip-gram. We explain these methods next.

Continuous Bag of Words (CBOW)

The CBOW model predicts the target word (w_t) based on the context words surrounding it [71, 70, 82, 67, 7, 100]. It uses a window of context words to predict the target word. The objective function for CBOW is to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_t|w_{t+j})$$

where T is the total number of words in the corpus, c is the size of the context window, and $p(w_t|w_{t+j})$ is the probability of the target word given the context words.

Skip-gram

The Skip-gram model, on the other hand, predicts the context words given the target word[71, 70, 82, 67, 26, 100]. It aims to maximize the average log probability of the context words given the target word:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

where T is the total number of words in the corpus, c is the size of the context window, and $p(w_{t+j}|w_t)$ is the probability of the context words given the target word.

Probability calculation

In both CBOW and Skip-gram models, the probability $p(w_O|w_I)$ is calculated using the softmax function [71, 82, 26]:

$$p(w_O|w_I) = \frac{\exp(\mathbf{v}_{w_O}\cdot\mathbf{v}_{w_I})}{\sum_{w=1}^{W}\exp(\mathbf{v}_w\cdot\mathbf{v}_{w_I})}$$

where $\mathbf{v}_{w_O}$ and $\mathbf{v}_{w_I}$ are the output and input vector representations of words w_O and w_I, respectively, and W is the vocabulary size.

3.6 Supervised learning

Supervised learning is a type of machine learning where the model is trained on a labeled dataset, meaning that each training example is paired with an output label [36, 47, 30, 99]. The goal is for the model to learn a mapping from inputs to outputs, enabling it to predict new, previously unseen (unlabeled) cases. Below, we describe common supervised models for multi-class (i.e., multiple possible output values) classification tasks available within the PySpark engine, which we will investigate as part of our research.

3.6.1 Supervised models

- **Logistic Regression**: Logistic regression is one of the most important statistical techniques used for binary classification problems [68]. It estimates the probability of a binary outcome based on one or more input variables, using the logistic function to ensure the predicted probabilities stay within the 0 to 1 range [52]. Logistic regression can also be extended to multi-class classification problems [68], which is particularly relevant in our case since we aim to predict multiple different usage types. In PySpark, this extension is supported through techniques such as multinomial logistic regression.

- **Decision Trees**: Decision trees are a widely used and highly effective data mining method for developing predictive models for a target variable [85]. They are versatile because both the target and independent variables can be either categorical or continuous, making them applicable in a variety of scenarios. A decision tree is composed of three types of nodes. The root node represents the initial choice that divides all records into two or more mutually exclusive subsets. Internal nodes represent possible decisions or events at various stages within the tree, linking parent nodes to child nodes. Finally, leaf nodes, or end nodes, represent the final predicted outcomes or classifications. The tree is constructed by recursively splitting the data at each node based on the feature that provides the best separation, according to criteria such as Gini impurity, which measures how mixed or impure data partitions are in terms of different classes, or information gain, which quantifies the reduction in entropy (a measure of disorder or unpredictability) achieved by the split [76, 85]. This process continues until the stopping criteria are met, such as reaching a certain depth or when further splits no longer significantly improve the consistency (homogeneity) of the data partitions [85].

- **Random Forest**: Random Forest is a regression tree method that employs bootstrap sampling and the random selection of predictors to create multiple decision trees, which are then combined to enhance predictive accuracy by reducing the noise and variability present in individual trees [80, 46]. A significant advantage of Random Forests is their ability to uncover complex interactions between variables, capture non-linear relationships and their applicability to both classification and regression tasks.

3.6.2 Evaluation metrics

To evaluate the performance of supervised models, it is common to use these standard metrics [74, 95, 99], which are also available within the PySpark engine:

- **Accuracy**: Accuracy is the ratio of correctly classified instances to the total number of instances. It is calculated as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

where TP is true positives, TN is true negatives, FP is false positives, and FN is false negatives.

- **Precision**: Precision measures the proportion of correct positive predictions out of the total number of positive predictions. It is calculated as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **Recall**:

  Recall, also known as sensitivity, is measured as the number of correctly classified positive predictions out of all positive instances. It is calculated as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **F1 Score**: The F1 score is a function of precision and recall that provides a balance between these two metrics. It is calculated as:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

These evaluation metrics provide a comprehensive assessment of supervised models' performance, enabling to select the most effective model for different classification tasks.

3.7 Unsupervised learning

Unsupervised learning is a type of machine learning algorithm that can learn from data without any prior knowledge of the outcomes or labeling [99]. In contrast to supervised learning, the focus in unsupervised learning is not on understanding the relationship between features and a target variable. Instead, it centers on identifying structures and patterns within the independent variables (non-target variables). This approach is particularly useful for us because we cannot fully verify the quality of the usage type attribute in the IP2Location dataset, and the methodology creation process is unknown. This makes it advantageous to perform the training phase based on the characteristics of the hostnames themselves, rather than the relationship between these characteristics and their labels, as is done in supervised learning.

Clustering is a common technique in unsupervised learning, which involves grouping similar data points into clusters [99]. In our research, we explore the potential application of this technique to group hostnames representing different usage types into separate clusters. We further discuss our methodology for inferring usage types using unsupervised learning in Chapter 6.

In the following subsections, we provide background information on different types of clustering algorithms and offer examples of commonly used algorithms that are available within the PySpark data analytics engine.

3.7.1 Types of clustering algorithms

There are a few types of clustering algorithms that are applicable to our study:

- **Centroid-based.** Centroid-based clustering is a method that organizes data into non-hierarchical clusters, where each cluster is represented by a centroid, which is the arithmetic mean of all the points within that cluster [79, 8]. Data points are assigned to clusters based on their distance to these centroids. Common distance metrics used in these clustering algorithms include Euclidean Distance, Manhattan Distance, Minkowski Distance and others [97]. In general, Centroid-based algorithms work particularly well when the resulting clusters are of similar sizes [8].
- **Density-based.** Density-based clustering algorithms group data points according to density objective functions, forming clusters in areas with high concentrations of data points [79, 8]. The main advantage of these algorithms is their ability to discover clusters of any shape while being robust to outliers.
- **Distribution-based.** Assumes that the data consists of various distributions. The probability of data points belonging to a specific distribution decreases as the distance from the center of the distribution increases [8].
- **Hierarchical-based.** Hierarchical clustering algorithms gradually build a tree-like structure of clusters, where each level of the tree represents a different level of similarity or grouping within the data. This method is particularly well-suited for organizing data that naturally forms a hierarchy, such as taxonomies [79, 8].

3.7.2 Clustering models

In this subsection, we discuss some common clustering algorithms from the aforementioned types that are also available in PySpark:

- **K-means (Centroid-based).** K-means is a centroid-based clustering method that uses a distance-based approach to partition data into distinct clusters [31]. The algorithm works by iteratively assigning data points to the nearest centroid and then recalculating the centroids based on the data points within the newly formed clusters. The goal of K-means is to minimize the distance between points within the same cluster, making them as similar as possible, while maximizing the difference to points in different clusters [99]. One of the main advantages of K-means is its relatively low computational complexity compared to other unsupervised algorithms, making it suitable for large datasets [31, 45, 79]. However, it also has notable disadvantages: the algorithm is sensitive to outliers and struggles with high-dimensional data, as K-means assumes spherical shape of clusters.
- **Gaussian Mixture Model (Distribution-based)** The Gaussian Mixture Model (GMM) is a probabilistic model that assumes that data points in different clusters are generated by different underlying Gaussian probability distributions [97, 28, 2]. This model is particularly powerful because it can capture complex, arbitrarily shaped data clusters by blending multiple Gaussian distributions, unlike K-means clustering, which relies on a single, fixed shape such as a sphere or ellipsoid.
- **Bisecting K-Means (Hierarchical-based)** Bisecting K-Means is a hierarchical clustering algorithm that combines the principles of k-means with hierarchical clustering [89]. It works by iteratively splitting clusters into two subclusters using the

K-Means algorithm, continuing this process until the desired number of clusters is reached. While Bisecting K-Means is efficient and performs well with large datasets, similar to K-Means, it shares the limitation of not being able to detect complex, non-spherical shapes within the data.

3.7.3 Clustering evaluation

In the following subsection, we discuss methods for evaluating clustering results. Since clustering is performed in an unsupervised manner, determining whether the results are good or bad is not straightforward, unlike in supervised analysis. This subsection focuses on methods that fall under internal validation, which involves using only the data that was clustered (no labels) [62, 91]. Additionally, we propose a method for validating results using external information (labels available), which is discussed in Chapter 6.

Among the common methods for internally assessing clustering results, one of those available within the PySpark data analytics engine is the Silhouette Score [83, 62, 6]. This metric measures how similar each data point is to its own cluster compared to the next closest cluster. The Silhouette Score ranges between -1 and 1, where -1 indicates incorrect clustering, 0 indicates mixed/overlapping clustering and 1 indicates dense, well-matched clustering.

Another simple and direct way to validate or check for any issues in clustering is by visually examining the results. One method is to create a plot showing the number of samples in each cluster, known as *Cluster Cardinality* [8]. This can reveal how the data is distributed across clusters and detect if one cluster is very different in size from the others, which, depending on the research goal, may or may not be a desired outcome.

3.8 Zero-shot learning

Zero-shot learning is a machine learning paradigm that enables models to perform classification tasks on categories without having seen labeled data for them during training [96, 77, 81]. This approach uses auxiliary information, such as descriptions or properties of the unseen classes, to make predictions without needing labeled samples. For our research, we consider the BERT zero-shot classifier available through the Spark NLP library, which is capable of large-scale data processing.

The BERT zero-shot classifier in the Spark NLP library is designed for zero-shot text classification, particularly in the English language [5]. The model is based on the BERT architecture, which is a bi-directional transformer encoder pre-trained on vast amounts of text data [58]. The core concept involves utilizing a model tailored for sequence classification tasks, where the model takes a sequence of text (in our case it can be hostname) and predicts a category (such as usage type) for the entire text. This capability is enabled by prior training on Natural Language Inference (NLI) tasks [5]. During NLI training, the model learns to determine the relationship between sentences, assessing whether one sentence entails, contradicts or is neutral to the other [86]. As a result, the BERT zero-shot classifier can infer relationships within textual data and predict categories for new text inputs without the need for further fine-tuning on specific tasks.

We integrate the BERT zero-shot classifier into our methodology to evaluate its effectiveness in accurately and efficiently categorizing Internet hosts without relying on comprehensive labeled datasets. This experiment aims to assess whether general-purpose pre-trained models can successfully handle such specialized data.

Chapter 4

Related work

In this chapter, we explore existing research focused on extracting meaningful information from hostnames. Given the challenges associated with obtaining high-quality and comprehensive IP data, which is often locked behind non-free commercial datasets, establishing the accuracy of findings becomes a critical challenge. Therefore, for each study, we will discuss not only the proposed research methods but also pay special attention to how the authors validated their results, particularly in the context of limited data availability.

Then, we describe and discuss research on feature extraction from hostname-like data, as earlier discussed in Chapter 3.

4.1 Hostname information extraction

There are multiple existing works that focus on the analysis of hostnames and domain names with the goal of extracting meaningful information from them. Luckie et al. discuss the design and implementation of a system that automatically learns regular expressions to extract the names of network operators embedded within hostnames [66]. By providing a complex but efficient method for generating these regexes, they achieved an accuracy of around 97%, verified through WHOIS and PeeringDB databases, which are commonly used to obtain registration and interconnection information for networks. As a core component of their research methodology, the authors relied on data from the CAIDA Internet Topology Data Kit (ITDK) to obtain Autonomous System Numbers (ASNs) for routers. The accuracy of this dataset is reported to be around 95%, based on comparisons with ground truth data from four network operators. In contrast, the IP2Location usage type data for IP addresses used in our study, both for training and testing of our models, has not yet been peer-reviewed by the research community. This reliance on proprietary, unverified commercial data represents a limitation of our work.

Another study, similar in concept and conducted by the same authors, focused on extracting router names (unique identifiers assigned by network operators, e.g., esr1|jfk2) from hostnames [63]. The authors employed a supervised learning approach to learn router naming conventions embedded in hostnames. They used extensive datasets spanning nine years, primarily drawing from the CAIDA Internet Topology Data Kit (ITDK) for training. The accuracy was relatively high, as network operators confirmed that 9 out of 11 suffixes' naming conventions were correctly identified.

There was also a study focused on extracting Autonomous System Numbers (ASNs) from hostnames [65]. This study aimed to identify the AS that operates each router by developing a system capable of automatically learning regular expressions (regexes) to extract these ASNs. Similar to previous research, the authors utilized training data col-

lected over approximately 10 years to create a set of reusable regexes. Their approach was evaluated against ground truth data provided by network operators and using PeeringDB database, achieving high accuracy. Consequently, they publicly released their training data and regexes.

The field of extracting geographical information from hostnames has been the focus of several studies [64, 37, 51, 84]. In their study, Luckie et al. [64] developed regular expressions (regexps) to extract geographical information from hostnames, significantly contributing to the field by leveraging the Hoiho regex builder. As part of their methodology for extracting geolocation information, the researchers utilized IATA and ICAO codes, standardized codes used to identify airports worldwide. These codes, sourced from OurAirports database, included corresponding latitude and longitude coordinates, which were instrumental in accurately mapping specific locations embedded in hostnames. Additionally, CAIDA's Internet Topology Data Kit (ITDK) was used to provide router-level graphs and hostnames for router interfaces. The ITDK datasets, which include both IPv4 and IPv6 routers, were crucial for training the system to recognize geohints and validate the extracted locations against real-world data.

As a result of their research, Luckie et al. achieved significant accuracy improvements in geolocating routers based on hostnames. Their system correctly identified 78.6% of the custom geographic codes created by network operators and accurately geolocated 94% of router hostnames that contained geohints. This success was particularly notable in cases where geohints were embedded in the hostname, as the system effectively deciphered these patterns to map locations. However, it is important to note that this high accuracy depended on the presence of recognizable geohints. While their system managed non-standard codes through the use of extra RTT measurements, this added complexity. In contrast, our approach relies solely on hostname analysis, without the need for additional measurements. Although our RQ3 focused on inferring geolocation as a validation point for our methodology in determining IP address usage types, the ability to extract geolocation from hostnames using methods beyond regex could have broader applications and better generalization. Even without apparent geohints, hostnames may exhibit patterns or characteristics common to specific network operators, particularly those operating in certain regions. By identifying and generalizing these patterns through manual or automated feature engineering, our system could enhance its ability to infer geolocation with greater accuracy, even in hostnames lacking obvious geohints.

The research conducted by Dan et al. diverged from previous research by specifically focusing on extracting geographical information from hostnames through the use of word feature extraction and machine learning techniques, thereby distinguishing it as particularly relevant to our research [37]. In their methodology, the authors implemented a comprehensive feature extraction process that involved dissecting hostnames into their elemental parts and generating both primary and secondary features based on city names, abbreviations, administrative regions and top-level domains. These extracted features were then combined with ground truth labels of known hostname locations, provided by Microsoft, and applied to machine learning models in a supervised learning approach. The method proposed by Dan et al. showed significant improvements over previous techniques in this domain, demonstrated through testing on a dataset of 1.6 million hostnames. This dataset was a subset of a larger ground truth dataset derived from Bing query logs, consisting of IP addresses with known locations, focusing on various domains from different Internet Service Providers (ISPs). The 1.6 million subset was specifically chosen to include entries from ISPs that were covered by multiple academic baselines, allowing for a fair and direct comparison between their method and the existing state-of-the-art techniques.

However, a limitation of this approach is the assumption that a single classifier should be universally applicable to all hostnames and remain effective over time, without accounting for the dynamic nature of hostname structures and the evolution of the internet. We believe this limitation does not pose a significant problem for our research, as we employ a combination of manually crafted features and features extracted using Word2Vec. While manually crafted features capture general, stable characteristics of hostnames, providing some resilience against change, the Word2Vec model's ability to learn a vast vocabulary and generate meaningful representations for previously unseen hostnames effectively mitigates this issue. This is because Word2Vec can recognize and interpret new hostname structures by leveraging its understanding of previously seen words within hostnames, allowing it to provide mostly accurate representations even when encountering new combinations or patterns.

In another study, Chabarek et al. [34] analyzed the DNS names of network device interfaces to identify common naming conventions among network operators. Although these DNS names are not strictly hostnames, they often bear similarities and offer valuable insights through their naming patterns. Authors utilized regular expressions to discern meaningful patterns from these names. Subsequently, they employed clustering techniques to categorize similar names, aiding in the identification of prevalent naming conventions. To confirm their findings, they compared these conventions with existing ones and conducted a survey with over 10 network operators to understand their naming practices. Their research revealed a diverse range of naming conventions used by network operators, with some being widely adopted and others more unique. The key takeaway from this study for our research is the substantial information can be derived from hostnames/domain names, particularly through the use of clustering techniques and word analysis.

In many of the studies we reference, authors commonly utilize regular expressions and external datasets to identify relevant words or phrases within hostnames. These methods often focus on extracting directly available information, as discussed in the research by Luckie et al. [64], such as extracting airport codes, which are standardized by organizations like IATA and ICAO. However, in our case, there is no standardized set of rules or dictionary that can reliably determine whether a hostname containing specific wording belongs to a particular usage type. Therefore, we employ manual and automatic feature engineering instead of relying on regular expressions, as we cannot directly extract the information or results we need. Moreover, our hostname dataset may include entries without any direct hints for inferring usage types. However, by employing feature engineering instead of regular expressions, our approach still allows us to predict usage types, even for hostnames that may initially seem to be completely uninformative.

Another significant difference is the scale of our data, which is substantially larger than those used in the aforementioned studies. This larger dataset enables us to conduct a more comprehensive analysis and achieve a more accurate assessment of our methods. For instance, Luckie et al. [64] analyzed a few million hostnames associated with IP routers that contained apparent geohints. In contrast, our analysis encompasses over a billion hostnames associated with a broader selection of IP addresses, not only limited to IP routers (see Chapter 5 for data description).

4.2 Feature extraction from hostnames

In a study referenced in *Towards Data Science*, the process of extracting features from URL strings for the detection of malicious URLs is examined [11, 33]. Despite the divergence

in subject matter, the article effectively highlights various lexical features extractable from URLs, including hostnames as a key component. We list the features applicable to URLs, which we expect to be applicable to hostnames as well. We will investigate this further and report on our findings in Chapter 6.

1. Total number of characters in the URL.
2. Quantity of digits within the URL string.
3. Shannon entropy of the URL string.
4. Count of `.` (dots) in the URL string (or other special characters).
5. Presence of specific keywords in the URL.

The authors emphasise that the features to be extracted are contingent upon the focus of the analysis. As their study centers on URLs, they extract a broader range of features pertinent to URLs, aligning with their objective of identifying malicious URLs.

In another study by Yadav et al. on detecting algorithmically generated malicious domain names, they mentioned two important ideas that could be useful for our research on hostname analysis [98]. First, analysing the frequencies of alphanumeric characters, on which many features in our case can be built. Second, conducting bigram analysis, which means not only analysing single characters but also pairs of consecutive characters.

Chapter 5

Data labeling, preprocessing and validation

5.1 Data labeling

One of the main goals of our research was to classify IP addresses by usage types based on associated hostnames. While usage type data is rare and often expensive, it does exist, with IP2Location providing such data. Our aim was to develop an open methodology that would be publicly available for other researchers to validate and build upon, in contrast to the closed (publicly unavailable) methodology of IP2Location. Still, since we needed ground truth for our supervised, unsupervised and zero-shot learning models, we made use of IP2Location usage type dataset [14].

Additionally, the IP2Location dataset included the country attribute, with its accuracy confirmed by other researchers to be close to 100% [57]. This attribute was particularly useful for us because most of the IP2Location data was not validated in terms of accuracy. The verified accuracy of the country attribute provided an additional way to validate our approach. By applying the methodology we developed for usage type prediction to geolocation country prediction, we could check whether we achieved similar results to the claimed 99-100% accuracy for the country attribute. This would suggest whether our usage type prediction approach was potentially good or not, given the lack of peer-reviewed reference points for usage types. The prediction of the country attribute was considered as our Research Question 3 (RQ3).

We received the IP2Location datasets in weekly dumps for the year 2023, which was important for us since we aimed to inspect how IP addresses' usage types changed over time (RQ2). To address RQ1, we selected the IP2Location dataset from the first week of January 2023, beginning with the first dump after January 1. This dataset consisted of 16,949,870 IP ranges, each linking a specific IP range to its corresponding usage type and country location.

As outlined in the documentation, the usage type column includes 12 possible categories for each IP range (see Table 5.1). Although we did not have control over the categories defined by IP2Location, these 12 categories comprehensively describe the various functionalities that IP addresses may be used for, making them sufficient for our research needs.

For the IP-hostname data for our research, we used the data obtained via reverse DNS measurements collected by the OpenINTEL project at the University of Twente [93]. These measurements are performed daily and cover the publicly routable portion of the address space. To investigate RQ1, we selected daily measurements from January 1, 2023. The number of obtained PTR records via reverse DNS for January 1, 2023, was 1,311,233,437 (≈

Usage Type	Description
COM	Commercial
ORG	Organization
GOV	Government
MIL	Military
EDU	University/College/School
LIB	Library
CDN	Content Delivery Network
ISP	Fixed Line ISP
MOB	Mobile ISP
DCH	Data Center/Web Hosting/Transit
SES	Search Engine Spider
RSV	Reserved

TABLE 5.1: IP2Location usage type categories

30% of the total number of IPv4 addresses). Of these, 1,285,834,541 records had hostnames (≈ 98% of the total PTR records obtained for January 1, 2023). Since our research relied on interpreting hostnames, we excluded those entries without hostnames for our work. Additionally, we counted a total of 6,138,963 IP addresses that were associated with more than one hostname, resulting in a total of 8,488,902 entries being duplicates in terms of IP addresses. Further in our research, we discuss how we handled duplicate entries for IP addresses, as these factors had different impacts on answering RQ1, RQ2 and RQ3.

The next step was merging the IP ranges data from IP2Location with individual IP-hostname entries in the OpenINTEL dataset. For this, we used py-radix tree structures to efficiently map IP ranges to their usage types and country, enabling fast lookup. We then performed these lookups on the IP-hostnames reverse DNS dataset. This process resulted in a total of 1,285,834,541 labeled samples (full label coverage of obtained reverse DNS data for January 1, 2023). The table below shows the count for each category in our labeled samples (see Table 5.2):

Usage Type	Count
SES	237,874
COM	34,757,786
MOB	97,502,699
ORG	1,765,961
EDU	26,513,504
ISP	377,934,555
LIB	70,707
CDN	15,018,356
DCH	156,547,412
GOV	4,360,933
ISP/MOB	566,047,362
MIL	5,075,225
RSV	5

TABLE 5.2: Count of labeled samples by usage type

It is interesting to note that our dataset included an additional category in contrast to the outlined IP2Location documentation: ISP/MOB, which likely represents IP ranges

that operate as both ISP and mobile providers. Additionally, we obtained only 5 IP addresses for the reserved (RSV) category. Due to the small size of this category compared to other usage types, we excluded it from our research. In total, we still had 12 usage type categories.

For the analysis over time, we selected four reverse DNS data snapshots: January 1, April 1, August 1 and December 1 of 2023, and performed a similar labeling procedure as for January 1. We describe the specifics of merging and processing these data snapshots for analysis over time later in Chapter 7.

5.2 Data verification

We mentioned that the accuracy of the usage type column from the IP2Location dataset is unverified by research community. Therefore, we conducted some verification checks on this column to assess its accuracy. To achieve this, we utilized several smaller datasets from large companies that publish their IP ranges. This allowed us to compare IP2Location's labeling with these publicly available ranges, from which we could infer usage types. For instance, Microsoft or Amazon ranges most likely fall within the datacenter (DCH) category. By comparing these IP ranges with the IP2Location data, we assessed how well IP2Location's labeling matches independent data. The selection of companies was designed to cover multiple categories within the IP2Location dataset while ensuring that we included some large companies and institutions.

Amazon

Firstly, we checked the Amazon IP ranges that they officially publish on their website [4]. For these IP ranges, which describe various services of Amazon, we performed a match with IP2Location data on an exact range match (inner join). The reason for conducting inner join on IP ranges was to simplify calculations, but still allowing us to verify a significant amount of IP ranges. The results of the service types of these IP ranges are shown in Table 5.3.

Type of Service	Count
DCH	3121

TABLE 5.3: Classification of Amazon IP Ranges according to IP2Location dataset

As can be seen, all matched IP ranges were classified into the DCH category, which stands for datacenters. This met our expectations, as DCH is one of the categories that Amazon IP ranges should definitively fall into.

Microsoft

We then assessed the IP ranges published by Microsoft and performed an inner join with IP2Location data [17]. The results are depicted in Table 5.4.

Type of Service	Count
ISP	55
DCH	13169
SES	105

Table 5.4: Classification of Microsoft IP Ranges

Similar to Amazon, most of the IP ranges fall into the category of data centers (DCH). A few fall into ISP and Search Engine Spiders (SES). These results are expected, as the majority fall into DCH, while the others likely reflect the diverse services Microsoft operates. The SES category most likely relates to the Bing search engine owned by Microsoft.

Verizon

Next, we analyzed the IP ranges available on a popular website tool called IPinfo [10], which allows lookup for IP address ranges of various companies. We searched for Verizon's IP ranges and then matched them with the IP2Location dataset. The results are shown in Table 5.5.

Type of Service	CIDR
ISP/MOB	103.22.238.0/23
ISP/MOB	132.197.217.0/24
ISP/MOB	132.197.219.0/24
ISP/MOB	132.197.220.0/24
ISP/MOB	132.197.234.0/24
ISP/MOB	132.197.235.0/24

Table 5.5: Classification of Verizon IP Ranges

As shown in Table 5.5, the IP ranges from Verizon were classified under the ISP/MOB category, which includes both ISP and mobile services. This result aligns with our expectations, as Verizon is known to provide both types of services.

Cloudflare

We also analyzed the IP ranges reported by Cloudflare [15], as shown in Table 5.6.

Type of Service	CIDR
CDN	103.31.4.0/22
CDN	131.0.72.0/22
CDN	197.234.240.0/22

Table 5.6: Classification of Cloudflare IP Ranges

As can be seen, these IP ranges from Cloudflare were classified under the CDN category, which stands for Content Delivery Network. This result is expected, as Cloudflare is primarily known for providing CDN services.

MIT

To explore different usage type category, we then found the IP ranges of MIT, a university that would likely fall under the EDU category [23]. The inner join operation results with IP ranges published by MIT are summarized in Table 5.7.

Type of Service	CIDR
EDU	18.0.0.0/11
EDU	128.52.0.0/16
EDU	129.55.0.0/16
EDU	198.125.176.0/20

Table 5.7: Classification of MIT IP Ranges

As can be seen, the IP ranges from MIT were classified under the EDU category. This result aligned with our expectations, as MIT is an educational institution.

VPN Experiment

Linking back to Chapter 1, to the example of Netflix fighting with VPN providers to block their IP ranges, we found some curated VPN range lists [22] and checked which categories they would most likely fall into. The results are depicted in Table 5.8.

Type of Service	Count
DCH	1656
ISP	74
COM	6
MOB	2
ISP/MOB	3
EDU	1

Table 5.8: Classification of VPN IP Ranges

As can be seen, most of the IP ranges fall into the DCH category, indicating that VPN addresses are potentially classified as data centers in the IP2Location dataset. Interestingly, during our research, we found forum discussions where ISP providers complained that they could not reach IP2Location developers to change their usage type to ISP from DCH, as many of their customers were being blocked by Netflix. While the reliance of companies like Netflix on IP2Location data hints at its utility, extensiveness and uniqueness, this alone does not conclusively confirm its accuracy. Further objective analysis is required to evaluate the precision and reliability of IP2Location usage type data comprehensively.

5.3 Data preprocessing

We analyzed different text preprocessing techniques discussed in Chapter 3 on the obtained and labeled dataset. We carefully evaluated the applicability of these techniques to determine the most effective methods for potentially improving the accuracy of predictions.

Noise removal

It is challenging to argue that noise removal is possible since hostnames do not have a standardized structure. For instance, if a hostname is fully or predominantly numerical or contains numerous digits, does this imply that it lacks utility and should be excluded? What other criteria might indicate that a hostname is invalid?

During our research, we observed that certain categories of hostnames, such as those labeled as ISP, MOB, ISP/MOB or CDNs, tend to contain numerical characters more

frequently than others. Hence, this indicates that discarding any data, even though it may seem logical at first, could be too risky.

Tokenization

As discussed earlier, the most common delimiter for tokenization is the `.` symbol. We used tokenization for various steps in our research. For instance, Word2Vec required words as input rather than whole sentences (in our case, the entire hostname), so we tokenized the hostnames, considering each part separated by `.` as a word. Other instances where we tokenized hostnames included creating manual features and using the BERT zero-shot classifier, which also expected tokenized text input.

Capitalization

In our case, we converted all hostnames to lowercase to ensure consistency. While DNS is case-insensitive and does not differentiate between uppercase and lowercase letters, it is possible that network operators might use capitalization to convey specific meanings. Therefore, although we standardized all characters in hostnames to lowercase for uniformity and simplified processing, it is important to note that this approach may overlook certain semantic nuances introduced by capitalization, though this is unlikely to cause significant issues.

Abbreviation

The use of abbreviations in hostnames presents a challenge due to the sheer volume of data and the lack of exploration of predicting usage types of IPs based on hostnames in previous public research. Despite this, we found no immediate need to specifically address abbreviations in our research. We utilize word embeddings, such as Word2Vec, which should inherently capture and categorize abbreviations commonly found in hostnames. Therefore, while abbreviations are not explicitly addressed in this study, they could be explored in future research using additional external datasets.

Stemming

Stemming, which reduces words to their base or root form by removing suffixes, is typically used in natural language processing to treat words with similar meanings as the same. However, for hostname data, stemming is not applicable. Hostnames often include specific prefixes, suffixes, and combinations that are meaningful in their entirety. Applying stemming could remove crucial parts of the hostname, leading to loss of information and misinterpretation of the data, especially for manual features. Additionally, hostnames frequently consist of technical terms, abbreviations, and codes that do not benefit from conventional stemming processes.

Lemmatization

Lemmatization, which reduces words to their base or dictionary form, is another technique used in natural language processing to handle different forms of a word. Similar to stemming, lemmatization is not applicable to hostname data. Hostnames are composed of a specific sequence of characters and segments that are meaningful only in their exact form. Altering these segments through lemmatization would distort the actual hostname and

could result in significant loss of information. Additionally, hostnames do not have variations in form that lemmatization is designed to address, making this process unnecessary and potentially harmful for accurate analysis and model training.

Chapter 6

Prediction of usage type (RQ1)

After labeling, validating, and preprocessing our data, we moved on to answering our research questions. We conducted our research using PySpark, a Python API for Apache Spark that enables large-scale data processing [18]. This engine provides numerous functions and methods that simplify the development and execution of efficient code. PySpark's SQL library, with its rich set of built-in functions for data manipulation and transformation, was crucial for many steps of our research. Additionally, Apache Spark's MLlib library offers a comprehensive suite of tools for implementing supervised and unsupervised learning models, as well as automatic feature extraction with Word2Vec and other necessary models for our research.

As discussed in Chapter 3, features derived from hostnames can be categorized into two main types: manual features and automatic features. Manual features are crafted based on domain knowledge with the intent to identify and distinguish specific characteristics within the data. For our study, we adopted manual features from relevant existing literature on URL analysis (see Section 4.2). These features were originally designed for URL strings, but because hostnames are a key component of URLs, we believed they could be relevant to our research and, therefore, incorporated them into our methodology.

The authors of the research on URL analysis emphasized that the features to be extracted should be contingent upon the focus of the analysis. Since our focus was on hostname data, we developed additional manual features alongside the adapted ones, leveraging our knowledge of hostnames. These new features were designed to capture their general characteristics, such as structural patterns and overall composition, while also addressing the potential limitations of tools like Word2Vec, which, although effective at identifying specific patterns and vocabulary, may not fully capture these broader, generalized aspects of hostnames.

6.1 Manual model features

Below, we provide a list of the features we crafted and discuss their implementation in PySpark for clarity:

Length

Calculates the total number of characters in the hostname. This feature was adapted from previous research discussed in Chapter 4, where the feature was the total number of characters in the URL. Since we have hostname data instead of URLs, our feature measures

the total number of characters in the hostname. We utilized a built-in *length* function of PySpark SQL for this calculation.

Number of digits

Calculates the total number of digits in the hostname. This feature was adapted from previous research discussed in Chapter 4, where the feature was calculated as the number of digits in the URL. Since we have hostname data instead of URLs, our feature counts the number of digits in the hostname. We utilized the built-in *regexp_replace* function to remove non-digit characters and then calculated the length of the resulting string using the *length* function of PySpark SQL library.

Character diversity

Measures the number of unique characters in the hostname. This feature captures the diversity of characters within hostnames, which can reflect different patterns or complexities. We implemented this by first splitting the hostname into an array of individual characters, then using the *array_distinct* function to remove duplicates and retain only unique characters. Finally, we counted the number of unique characters using the *size* function in PySpark SQL.

Hyphen count

Counts the number of hyphens in the hostname. This feature helps to identify the presence of specific patterns or naming conventions within hostnames. We implemented this by calculating the difference between the total length of the hostname and the length of the hostname after removing all hyphens using the *regexp_replace* function in PySpark SQL. The resulting difference represents the total number of hyphens in the hostname.

Shannon Entropy

Calculates the Shannon Entropy of the hostname characters, reflecting the randomness or unpredictability of the character distribution within the hostname. This feature was adapted from previous research discussed in Chapter 4, where Shannon Entropy was calculated for URL strings. Since our dataset consists of hostname data rather than URLs, we applied the same entropy calculation to measure the diversity of characters in each hostname. The Shannon Entropy is computed by determining the frequency of each character in the hostname, calculating the probability distribution of these characters, and then summing the weighted logarithms of these probabilities. We implemented this using a custom user-defined function (UDF) that was applied to each hostname in our dataset.

Special character count

Counts the number of special characters (non-alphanumeric) in the hostname. This feature helps to identify the presence of special characters that might indicate certain patterns or specific naming conventions within hostnames. We implemented this by removing all alphanumeric characters and periods using the *regexp_replace* function in PySpark SQL. The remaining characters are then counted to determine the number of special characters in the hostname.

Vowel count

Counts the number of vowels in the hostname. This feature captures the frequency of vowel usage, which could reveal underlying patterns or naming conventions within hostnames. We implemented this by using the *regexp_replace* function in PySpark SQL to remove all non-vowel characters from the hostname, and then calculating the length of the resulting string to determine the total number of vowels.

Consonant count

Counts the number of consonants in the hostname. This feature captures the frequency of consonant usage, which could provide insights into the structural characteristics of hostnames. We implemented this by using the *regexp_replace* function in PySpark SQL to remove all non-consonant characters from the hostname, and then calculating the length of the resulting string to determine the total number of consonants.

Vowel to consonant ratio

Calculates the ratio of vowels to consonants in the hostname. This feature may provide additional insights beyond separate vowel and consonant counts by normalizing the relationship between these two metrics. It could highlight linguistic patterns, such as whether a hostname tends to favor vowel-heavy or consonant-heavy constructions. We implemented this by dividing the already calculated vowel count by the consonant count.

Token count

Calculates the total number of tokens (segments) in the hostname, where tokens are parts of the hostname separated by dots (`.`). As discussed in Section 3.2, we previously explained that the dot is a natural delimiter for hostnames, as it effectively separates the hierarchical components of a domain name. This feature was adapted from previous research discussed in Chapter 4, where the count of dots in the URL string was used as a measure of the complexity of domain names in URLs. We implemented this feature using the *split* and *size* functions from PySpark SQL. The *split* function is used to break the hostname into segments based on the dot delimiter, and the *size* function counts the resulting segments.

Repeated character ratio

Computes the ratio of characters in the hostname that are repeated. This feature captures the frequency of recurring characters, which can provide insights into the patterns and consistency within the hostname structure. High ratios might indicate hostnames with repetitive patterns. This feature was implemented by first iterating over each character in the hostname to count the number of occurrences of each character. We then identified the characters that appeared more than once and incremented a sum by 1 for each character that was duplicated. Finally, we divided this sum by the total number of characters in the hostname to calculate the ratio. This calculation was implemented using a custom UDF and functions from the PySpark SQL library to efficiently process each hostname in our dataset.

To ensure the relevance and effectiveness of our manual feature engineering, we conducted an ANOVA F-test to evaluate the potential connection between the crafted features,

including those adopted from previous research as well as the ones we independently designed, and the usage types of hostnames.

ANOVA F-test is appropriate for situations where the features are numerical and the target variable is categorical, as in our case. We selected a p-value threshold of 0.05, a common standard for statistical significance, to reject the null hypothesis and deem a feature relevant if its p-value was below this threshold.

For the ANOVA F-test, we used UnivariateFeatureSelector from Pyspark ML library. As a result of this test, we found that our 11 manually crafted features exhibit a significant association with the target variable. This suggests that these features effectively differentiate between the categories of the target variable, as the differences in their means across groups are notably larger than the variation within each group. Additionally, in Figure 6.1, we present the mean values of each feature across different usage types. As shown, many of these features, even from visual inspection, appear to significantly contribute to distinguishing between different usage types.

FIGURE 6.1: Mean values of each feature across different usage types

6.2 Supervised learning

After identifying the manual features for our machine learning models, we proceeded to predict the usage type of hostnames. In addition to the selected manual features, we incorporated automatic features generated by the Word2Vec technique, which produces vector representations of textual data, specifically hostnames in our case.

We employed a supervised learning approach, training our models using the available ground truth data. For this categorical prediction task, we utilized three supervised models from the PySpark ML package: Logistic Regression, Decision Trees and Random Forest.

The performance of these machine learning models, using both manual and automatic features as input, was evaluated using a single 80:20 train-test data split on a labeled data snapshot from January 1, 2023. This ratio (80:20) is a widely adopted practice based on empirical analyses showing that it tends to yield valid and reliable accuracy estimates [43]. The results are presented below:

Classifier	**Accuracy**	**Precision**	**Recall**	**F1 Score**
DecisionTreeClassifier	0.5786	0.6049	0.5786	0.5151
LogisticRegression	0.4955	0.4470	0.4955	0.4312
RandomForestClassifier	0.6192	0.5753	0.6192	0.5658

TABLE 6.1: Usage type prediction metrics using manual features

Classifier	**Accuracy**	**Precision**	**Recall**	**F1 Score**
DecisionTreeClassifier	0.6018	0.6638	0.6018	0.5400
LogisticRegression	0.6801	0.6904	0.6801	0.6648
RandomForestClassifier	0.6589	0.7043	0.6589	0.6177

TABLE 6.2: Usage type prediction metrics using automatic Word2Vec features

The metrics indicate that automatic features yielded better results overall. However, we observed a class imbalance in our data, where certain categories were underrepresented compared to others (refer to Chapter 5 for the distribution of categories). It is well known that classifiers tend to be more sensitive to the majority class, often overlooking the less frequent categories, making it challenging for models to accurately distinguish these minority classes [42]. That is why we plotted confusion matrices to assess whether the models could effectively identify and differentiate these minority classes despite the imbalance. Below, we present the confusion matrices for the models that achieved the highest accuracy and demonstrated the best overall usage type detection using separately manual and automatic features as input (see Figures 6.2 and 6.3).

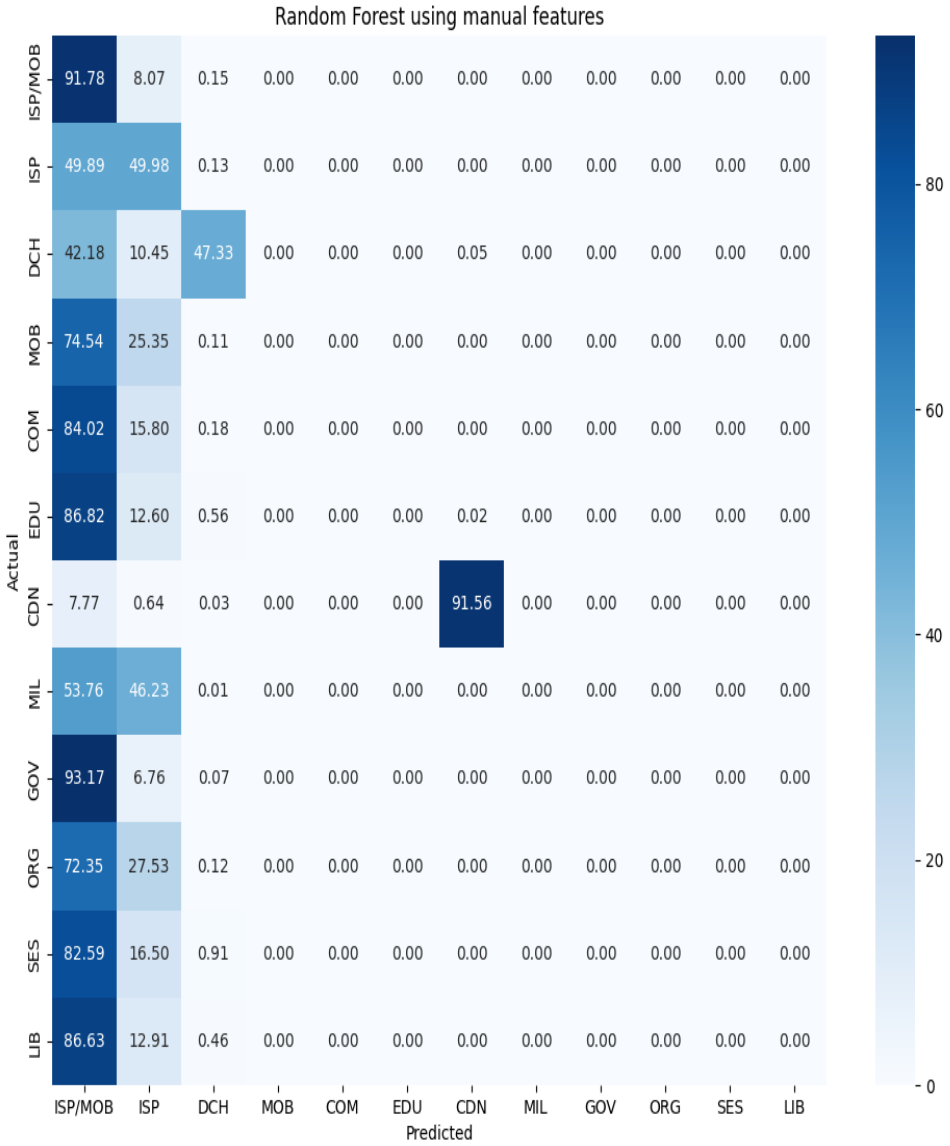

FIGURE 6.2: Random Forest model using manual features. The values within the matrix represent percentages of the total actual values in each row (for each category), making it easier to interpret large numbers by showing how they distribute across the predicted classes.

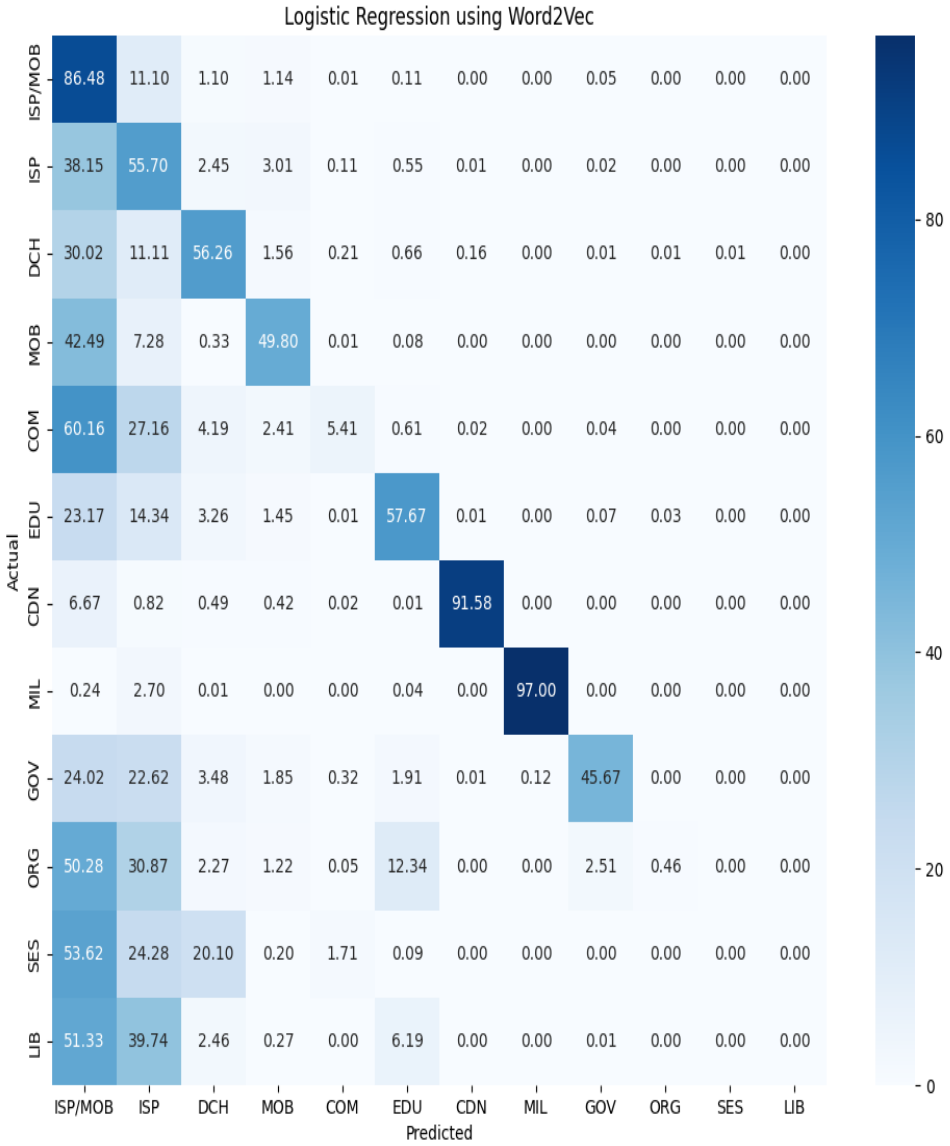

FIGURE 6.3: Logistic Regression model using Automatic (Word2Vec) features. The values within the matrix represent percentages of the total actual values in each row (for each category), making it easier to interpret large numbers by showing how they distribute across the predicted classes.

For manual features, the accuracy reached 60%, but the confusion matrix revealed that only ISP/MOB, ISP, DCH and CDN usage types were detected. For the other classes, the machine learning models did not generalize well and tended to classify most entries into ISP/MOB and ISP majority classes. Hence, manual features alone were insufficient for effective model learning for the hostname usage type classification task.

In contrast, using automatic features as input, the prediction of smaller classes became

more accurate, revealing that many usage type categories of hostnames could be detected. Among the three models, the logistic regression model performed the best, distinguishing multiple classes with the highest accuracy. Categories such as CDN (with similar accuracy to manual features), MIL and ISP/MOB were well detected. Notably, hostnames belonging to the military category were detected with 97% accuracy, suggesting that these hostnames share a similar vocabulary, making them easier to identify using automatic features. In contrast, MIL category was not detected at all with manual features as input.

The ISP, DCH, MOB, EDU and GOV categories were moderately well-identified, suggesting that the Word2Vec approach is more effective than manual feature selection. However, some classes, such as SES, COM, ORG and LIB, remained challenging to detect or were completely undetected. This difficulty could be attributed to the low number of samples in these categories, leading to an imbalanced dataset. Interestingly, despite having many samples, the COM category was also poorly detected, likely because COM hostnames share similar structures and naming conventions with ISP and MOB hostnames, leading to misclassification into these categories.

In summary, detection accuracy using automatic features was notable, despite the challenging nature of the task and the lack of standardized rules for defining hostnames by network operators. We think that this complexity is the main reason it is difficult to surpass very high accuracy in identifying usage types based on hostnames. Nonetheless, this approach can still be relevant in practice, as it eliminates the need for access to private datasets containing usage type information. Instead, one only needs to query the hostnames of IP addresses using reverse DNS and apply our trained model to infer usage types. This model should handle unseen hostnames using a predefined dictionary through Word2Vec. Categories such as ISP/MOB, CDN and MIL can generally be detected with very good accuracy.

Conversely, previous research on geolocation prediction from hostnames has successfully utilized rules (conventions) like common abbreviations and country codes to infer geolocation data. This success can be attributed to the limited number of countries and abbreviations, resulting in a more restricted and defined vocabulary, thus increasing the likelihood of higher accuracy. However, predicting usage type lacks these constraints, making it more challenging, especially given the data size. Later in our research, we explore predicting the country attribute based on hostnames using a similar approach we used for predicting usage type. Achieving high accuracy in geolocation prediction would suggest that the vocabulary list captured by the Word2Vec technique is in fact more standardized and defined for geolocation data.

It is also important to note that we performed limited parameter tuning for our classification models due to the long training times required for the large dataset we used. Therefore, we believe that further model tuning could potentially increase accuracy and improve the detection of different usage type categories. For instance, for Word2Vec, we chose the parameters *min_count=100* and *vector_size=50*. The *min_count* of 100 was chosen because of our dataset's large size, necessitating the exclusion of words that occur less than 100 times. Lowering this value could improve vectorization by increasing the dictionary size. A vector size of 50 was selected as a reasonable choice for the next prediction steps. Although literature suggests higher values for optimal performance [70], the large size of our dataset and memory limitations during the training phase necessitated this choice.

Furthermore, it is worth noting that Word2Vec averages the vector representations of words within the same document, which in our case corresponds to within the same hostname. This means that increasing the vector dimensionality could result in more

distinct vector spaces, thereby enhancing the separation between categories. As a result, with increased resources, optimizing the vector size and other model parameters could significantly boost both accuracy and detection capabilities.

6.3 Unsupervised learning

As we previously discussed, internet data is often limited and ground truth can be difficult to obtain and may lack proven quality. This challenge also applies to our usage type prediction. To address this, we explored unsupervised and zero-shot learning approaches. Unsupervised learning is advantageous because it does not rely on ground truth during training, instead focusing on identifying structures and patterns within hostnames. Meanwhile, zero-shot learning is beneficial as it does not require prior learning, making it a flexible tool for our research.

For the unsupervised learning approach, we investigated clustering algorithms to categorize Internet hosts based on hostname data. This experimental method aimed to manage the complexity of hostname data, which differs from typical textual data, by grouping similar data points into clusters. The objective was to distinguish hostnames without relying on extensive labeled data, instead utilizing a small number of samples to identify clusters corresponding to specific usage types.

For our analysis, we utilized the KMeans clustering algorithm as it provides good separation of data points and is reasonably fast for large datasets. Other models, such as the Gaussian Mixture Model (GMM) and Bisecting KMeans available in the PySpark ML library, took a very long time to train and were left out of the scope of this research.

For KMeans clustering, we set the number of clusters to 12 to match the number of distinct usage type classes we had. Our hypothesis was that, ideally, hostnames belonging to different categories would have distinct vector representations and would predominantly fall into their respective clusters.

To assess the results, we used the Silhouette Score, which we explained earlier in Chapter 3, and our custom visual inspection method, where for each cluster we plotted the distribution of usage types of data points that were assigned to that cluster.

For clustering, we utilized automatic features as input because they produced significantly better results in supervised prediction, particularly in detecting more categories. This indicates that these features better distinguish between different usage types, which suggests they may also enhance the differentiation of data points in unsupervised learning. We obtained the following results for each cluster:

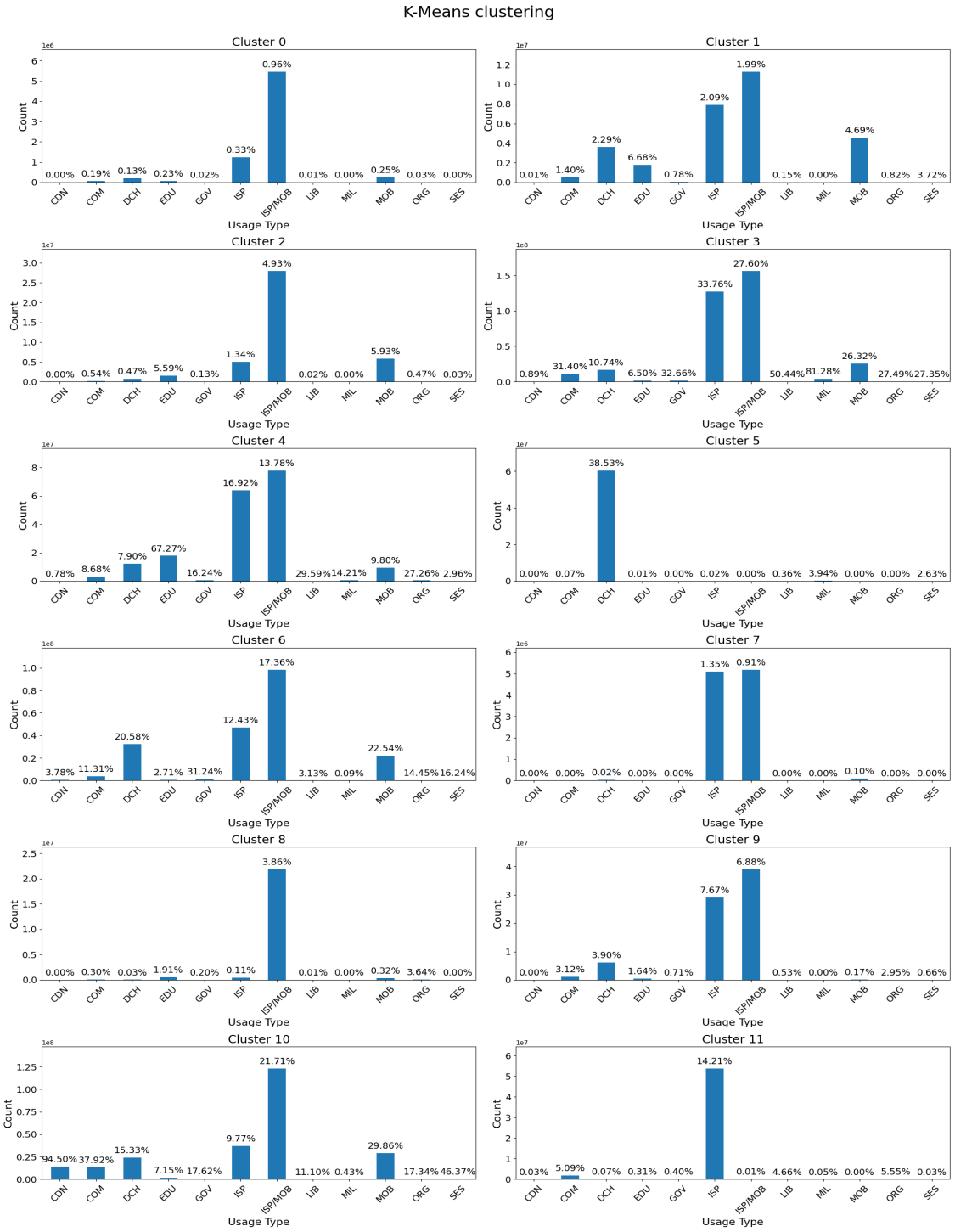

FIGURE 6.4: K-Means clustering using Word2Vec features as input. The figure shows the distribution of usage types across 12 clusters, with each cluster represented in a separate subplot. The height of each bar represents the raw count of a specific usage type within the cluster, while the percentage labels on top of each bar indicate the proportion of that usage type relative to its total occurrences across all clusters

As can be seen, the clustering results did not meet our initial expectations. Ideally, we anticipated that each usage type would be distinctly grouped within one of the 12 available clusters. However, as shown in Figure 6.4, there was a significant mix of usage types across different clusters. Interestingly, the DCH usage type class predominantly occupied Cluster 5, representing 38.53% of the total DCH IP addresses. We would have preferred this figure to be closer to 100% for DCH in Cluster 5, and similarly for other usage types in their respective clusters, as this was the intended goal.

The Silhouette Score we obtained was 0.227, indicating poor separation of data points, likely due to the limitations of K-Means in handling high-dimensional, complex data [31, 45, 79]. However, using Word2Vec-generated features with unsupervised learning could potentially yield better results with less complex, more separable data, particularly in small datasets where algorithms like Gaussian Mixture Models in PySpark could be applied.

Lastly, it is important to note that while we utilized all available ground truth data to verify usage types, in a real-case scenario, only a portion of the ground truth might be used to infer which category each cluster represents.

6.4 Zero-shot learning

Prior to our research, we experimented with using ChatGPT-4, an advanced language model developed by OpenAI, to infer the usage type of several hostnames. The results were somewhat promising, as the model mostly successfully categorized hostnames into one of 12 possible usage types.

At the time of our research, however, ChatGPT-4 was not feasible for handling the large-scale, distributed processing required for our extensive dataset. To handle the distributed processing for our large-scale data in PySpark, we considered using distributed class prediction tools like zero-shot classifiers. These tools, while conceptually similar to ChatGPT-4 in their ability to generalize across tasks, are specifically designed for classification based on a supplied list of categories, making them highly adaptable to various datasets. Given the unique nature of hostname data, it was intriguing to explore whether this approach, like our prior use of ChatGPT-4 for several hostnames, would yield promising results.

For our research, we applied a BERT zero-shot classifier to a limited set of 100,000 randomly selected data points due to issues in our setup of the Spark NLP library. The classifier provided probabilities for each hostname belonging to one of the 12 categories, and we selected the highest probability as the predicted output. However, we achieved only about 20% accuracy, likely because hostname data is not typical textual data and the BERT zero-shot classifier struggled with this complexity. Despite efforts to tweak parameters, input queries, and category lists, significant improvements in accuracy were not observed.

Hence, among the supervised, unsupervised and zero-shot learning approaches explored, the supervised method proved to be the most effective for inferring usage types from hostnames, achieving higher accuracy and more promising results overall.

Chapter 7

Usage type over time (RQ2)

To investigate trends in IP address usage over time (RQ2), we selected four data snapshots (four specific days) to observe shifts throughout the year:

- **January 1**
- **April 1**
- **August 1**
- **December 1**

Similar to RQ1, we labeled the IP-hostname datasets for April 1, August 1, and December 1 with the corresponding usage types. We limited our analysis to only four snapshots due to the time and resource-intensive nature of labeling over 1 billion IP addresses multiple times.

The next step involved merging these four datasets based on IP addresses to track usage type changes over time. Since some IP addresses in the reverse DNS measurements may appear or disappear over time [92], we merged these datasets only when an IP address was present in all four snapshots.

Additionally, as discussed in Chapter 5, we identified duplicated IP entries, indicating that some IP addresses were associated with multiple hostnames. For this research question, we decided to remove all IP addresses with multiple associated hostnames. This decision was made because accurately interpreting hostname changes over time in such cases would be challenging. Furthermore, merging datasets with multiple entries for the same IP address could create a Cartesian product, significantly increasing data size and causing computational challenges.

Hence, first, we examined the number of hostnames that changed over time to understand the scale of these changes. The observed dynamics were as follows:

Period	Hostname changes
January 1 - April 1	9,054,283
April 1 - August 1	13,162,719
August 1 - December 1	25,230,449

TABLE 7.1: Hostname changes over course of 2023

As shown, the number of changes was not significant given the full dataset of over 1 billion entries. However, an increasing trend was observed throughout 2023. For the

hostnames that changed over time, we also determined how many had changes in their assigned usage type. We obtained the following results:

Period	Hostname changes that resulted in usage type changes
January 1 - April 1	72,126
April 1 - August 1	455,317
August 1 - December 1	151,885

TABLE 7.2: Hostname and usage type changes over course of 2023

As a result, few hostnames that changed led to changes in usage type, and there did not appear to be a correlation between hostname changes and corresponding changes in usage type. This suggests that network operators do not necessarily update hostnames to reflect changes in usage types or that usage type is updated with some delays in the IP2Location dataset.

We also determined changes in usage type over the year based on changes in the usage type column in IP2Location dataset, regardless of whether the hostname changed or not. The results were as follows:

Period	Usage type changes over the year
January 1 - April 1	5,126,357
April 1 - August 1	17,918,038
August 1 - December 1	5,198,550

TABLE 7.3: Usage type changes over course of 2023

As can be seen, there were significantly more changes in the usage type of IP addresses based solely on changes in the usage type column in the IP2Location dataset than in hostname changes that resulted in a usage type change.

To gain deeper insights into answering our RQ2, we tracked the shifts in usage types across these four data snapshots and visualized the data using a Sankey diagram. The results are shown in Figure 7.1:

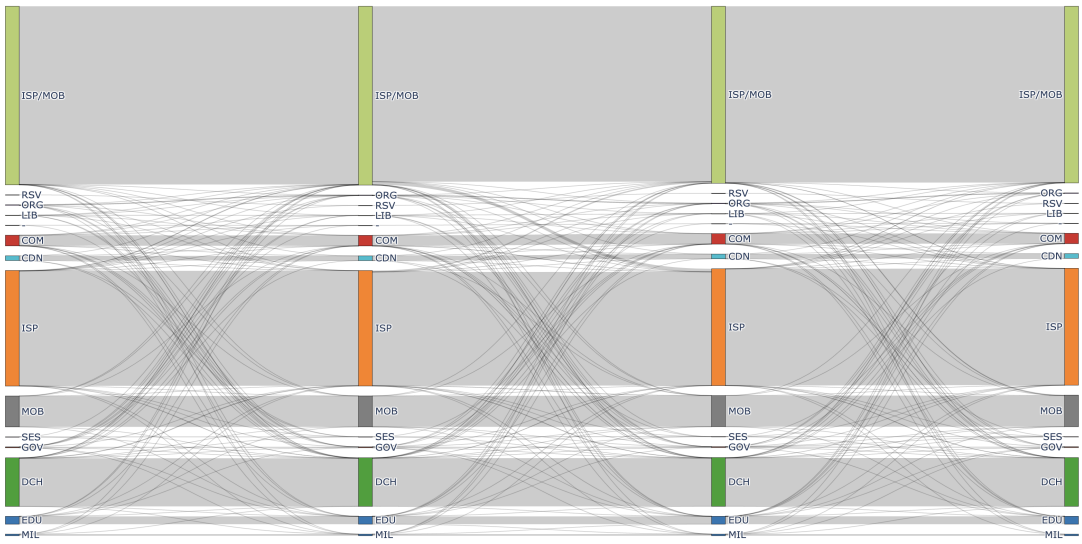

FIGURE 7.1: Usage type changes over time in 2023

As mentioned earlier, the number of changes in usage type is relatively small compared to the overall data. Therefore, we removed links of usage types to themselves in the next data snapshot (e.g., ISP to ISP) to better visualize the shifts. The results are shown in Figure 7.2:

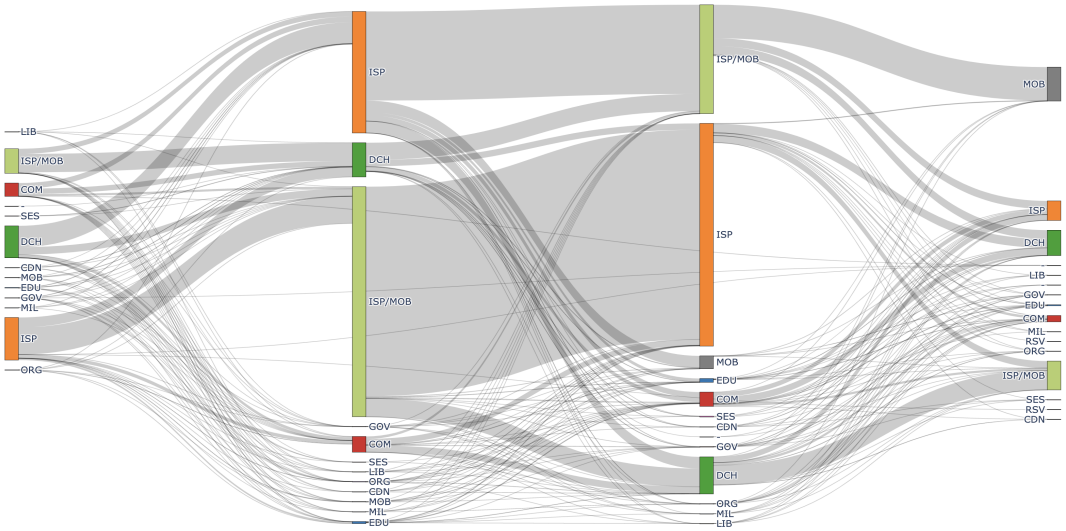

FIGURE 7.2: Usage type changes over time in 2023 (filtered to exclude self flows)

The transition data between different IP address categories from January to December 2023 reveals several interesting trends and movements across three distinct timeframes.

During the first timeframe, a significant shift was observed in the DCH (datacenter) category, with a large number of IP addresses transitioning primarily to ISP. Additionally, a notable portion moved to ISP/MOB and COM. This trend suggests a strategic reallocation of data center resources towards more versatile and accessible internet service providers and mobile networks, possibly driven by the increasing demand for robust connectivity solutions.

In the ISP/MOB category, there was a noticeable reclassification of addresses to DCH and ISP, indicating a refinement in how these services are being categorized and managed. The COM category saw substantial shifts towards ISP and DCH, reflecting a broader integration of commercial activities with core internet services and data center operations.

The ISP category experienced a diverse set of transitions, with significant movements to ISP/MOB, COM, and DCH, highlighting the dynamic nature of internet service management. Additionally, there was an influx of addresses into the EDU category from ISP, suggesting an expansion in educational networks and their connectivity infrastructure. Other categories did not exhibit significant changes during this period.

In the second timeframe, there were substantial switches between ISP and ISP/MOB, reflecting ongoing efforts to balance and optimize internet and mobile service delivery. There was also a considerable reclassification of IP addresses from ISP to MOB and DCH, indicating a shift towards more mobile-centric and data-focused infrastructure.

The ISP/MOB category saw significant movements to DCH, pointing towards a growing reliance on data center capabilities to support mobile network operations. The COM category continued to transition mainly to ISP and DCH, suggesting an ongoing integration of commercial networks with these critical infrastructure components.

DCH exhibited significant changes with large numbers of IP addresses moving to

ISP/MOB and ISP, underscoring the central role of data centers in supporting both internet and mobile services. Other categories remained relatively stable with no notable changes.

The final timeframe showed fewer changes compared to previous periods. However, there were notable transitions from ISP/MOB to MOB, reflecting a major reclassification towards mobile network addresses. Additionally, significant movements from ISP/MOB to ISP and DCH were observed, indicating a rebalancing of services.

The ISP category saw further transitions to DCH and ISP/MOB, continuing the trend of dynamic reclassification and optimization of internet services. DCH maintained its role as a central hub with significant movements to ISP and ISP/MOB, highlighting the importance of data centers in the evolving network landscape. The COM category continued to switch to ISP and ISP/MOB, further solidifying the interconnected nature of commercial and internet services.

Hence, throughout the year, the transition data reveals a dynamic and evolving landscape of IP address categorization. Significant trends include the strategic reallocation of data center, internet and mobile resources and the expansion of educational network infrastructure. These insights highlight the ongoing efforts to enhance connectivity, optimize resource utilization and adapt to the growing demands of a digitally interconnected world.

Additionally, we calculated the total values for each usage type in the previously defined time snapshots to get an overall picture of the increase or decrease in the volumes of different usage types over time (see Table 7.4):

	2023-01	2023-04	2023-08	2023-12
ORG	1,605,179	1,602,706	1,596,910	1,594,310
CDN	15,016,157	15,013,891	15,020,571	15,019,318
ISP/MOB	561,740,734	562,286,016	556,836,047	555,838,722
COM	34,330,360	34,082,040	33,901,703	33,541,518
DCH	155,874,840	155,974,225	156,346,381	156,221,314
ISP	375,693,044	375,242,294	379,794,862	379,807,236
SES	236,348	227,539	233,957	198,645
GOV	4,278,217	4,276,088	4,292,374	4,290,783
LIB	69,525	68,837	71,454	71,121
MIL	5,021,233	5,021,109	5,019,746	5,019,489
MOB	97,320,918	97,320,874	97,882,036	99,395,760
EDU	26,156,927	26,227,863	26,347,441	26,345,266

Table 7.4: Total of each usage type over the course of 2023

The consistent decline in the ORG category throughout the year suggests ongoing but slight organizational changes or continuous data reclassification. The CDN category remained stable throughout 2023, indicating steady demand for content delivery networks, with no major shifts in content distribution methods.

The overall decrease in ISP/MOB might reflect better classification into more specific categories or a stabilization in mobile data usage, possibly indicating a market shift. The steady decrease in COM suggests a consistent reduction, possibly due to changes in commercial data allocation or shifts in usage patterns.

The overall increase in DCH highlights a growing emphasis on data center operations, likely driven by the increasing need for data storage and processing. The increase in ISP over the year suggests a trend towards expanded internet service provision, possibly

reflecting market growth or infrastructure expansion.

The SES category remained relatively stable, with a slight decrease towards the end of 2023. The GOV, LIB, and MIL categories showed little change over the year, likely reflecting stable operations in these services and minimal need for reclassification.

The significant increase in MOB, especially in the later stages of 2023, underscores the growing importance of mobile data services in modern internet usage, likely reflecting the ongoing expansion of mobile networks. The moderate but consistent increase in the EDU category points to sustained growth in educational IP address usage, possibly driven by continued online learning and digital educational initiatives.

Lastly to mention, if high-accuracy predictors of usage type for any IP address can be developed, it would be possible to conduct this analysis over time without relying on the ground truth provided by IP2Location, but rather on model predictions. The only limitation here is that this analysis would depend on hostname changes, which would not provide a complete picture of usage type changes over time (based on findings above).

Chapter 8

Prediction of country attribute (RQ3)

To further validate our methodology for predicting usage types and assess its practical usability, we used the prediction of the country attribute as an additional validation point. Previous academic research had verified that IP2Location's country attribute has an accuracy close to 100%. Therefore, if our methodology achieves high accuracy for the country attribute, it could suggest that our approach also likely yielded practically usable results for predicting usage types, which is the primary focus of our research.

For this prediction, we utilized the methods that worked best for answering RQ1. Specifically, we used Word2Vec automated features and three supervised learning models: Decision Trees, Random Forest and Logistic Regression. We obtained the following results:

Classifier	**Accuracy**	**Precision**	**Recall**	**F1 Score**
DecisionTreeClassifier	0.4889	0.4051	0.4889	0.3853
LogisticRegression	0.9039	0.9084	0.9039	0.9017
RandomForestClassifier	0.5699	0.5916	0.5699	0.4841

TABLE 8.1: Country prediction metrics using automatic Word2Vec features

Similar to previous experiments, Logistic Regression yielded the best results. This outcome may be linked to the need for further tuning of other models to achieve higher accuracy or the possibility that the data is linearly separable, explaining why logistic regression performed well.

Notably, for the country prediction we achieved an accuracy of over 90%. This result was somewhat expected, as we discussed earlier, there is a limited number of countries, country codes and conventions. Consequently, Word2Vec was able to capture general patterns and common words along with their vector representations, making the prediction of this geolocation data highly accurate.

Last but not least, these high accuracy results highlight the potential of our methodology, which leverages Word2Vec for feature extraction and a supervised learning approach for usage type prediction. In Chapter 10, we further discuss strategies for enhancing accuracy in usage type prediction.

Chapter 9

Limitations

During the course of this research, several limitations were identified that may have impacted the generalizability and robustness of our findings. These limitations are discussed below:

Data coverage

Although our study analyzed over 1 billion IPv4 addresses, it is important to acknowledge that not all hosts had PTR records obtainable via reverse DNS. This absence could have introduced a blind spot in our analysis, potentially affecting the generalization of our results. Nonetheless, we believe that the vast quantity of data processed, particularly when compared to other studies on Internet data, sufficiently validated our methodology. However, this limitation remains a factor in the overall interpretation of our study.

Reliance on IP2Location data

Our research relied heavily on the IP2Location commercial dataset as the ground truth for the usage type attribute. The accuracy of this dataset, especially of usage type information, had not been independently verified by other researchers, introducing a potential limitation in our study. Additionally, we had no control over the classification categories provided by IP2Location. These categories were predefined, and while we believe they comprehensively cover the IP address space with various distinct usage types, our dependence on this dataset remains a limitation.

Impact of data preprocessing on accuracy

Hostname data, unlike conventional textual data, do not adhere to standardized linguistic rules, presenting a significant challenge in our analysis. In this research, specific choices were made regarding which preprocessing techniques to apply and which to exclude, with justifications provided for these decisions. However, this remained a limitation, as our choices could have benefited from further accuracy comparisons on prediction accuracy to fully understand the impact of including or excluding certain preprocessing techniques. Additionally, PTR records, which are not sanitized and can contain nonsensical data, were not thoroughly analyzed for noise removal during the preprocessing phase. This limitation could have impacted the accuracy of our predictions, likely decreasing it.

Chapter 10

Future work

In the following chapter, we present two key directions for future work that could build upon our research.

Improving prediction accuracy

One important area for future work is enhancing the accuracy of usage type prediction. This can be pursued through several approaches:

- **Enhancing Word2Vec features:** Increasing the memory allocated to Word2Vec features could lead to better results. Specifically, expanding the size of the word vectors and lowering the minimum count threshold could create a more detailed dictionary of word representations. These changes could make the averaged vectors for hostnames more distinct and separable across different usage type categories, leading to better prediction accuracy.
- **Exploring more complex feature extraction methods:** Another promising approach is to explore more advanced alternatives to Word2Vec for automatic feature extraction. For example, the Spark NLP library [20] offers BERT word embeddings, which, although more resource-intensive, could potentially yield better results due to their complexity. Exploring these advanced methods could significantly improve the accuracy of usage type predictions by enabling the creation of more nuanced vector representations.
- **Incorporating additional data:** Future research could also consider adding more attributes or using external data sources. For instance, integrating data from WHOIS databases, peering databases, or other public IP address sources could provide more context, helping the model to better differentiate between usage types. While this study purposely focused only on hostnames, expanding the dataset to include these additional sources could offer valuable new insights.

Expanding the time frame of analysis

For RQ2, we analyzed reverse DNS data from 2023, but future research could benefit from extending this analysis over multiple years. A longer study period would allow to observe long-term trends and shifts in usage types, offering deeper insights into how Internet usage patterns evolve over time. This would require access to more historical data and the ability to process and analyze larger datasets.

Chapter 11

Conclusion

In this research, our primary goal was to develop an open methodology for predicting the usage types of IP addresses based on their associated hostnames. We explored a range of approaches, including supervised, unsupervised and zero-shot learning, to address the complexities inherent in this task. The rationale behind incorporating unsupervised and zero-shot learning was to experiment with methods that do not rely on ground truth data during the training phase, particularly since we relied heavily on the IP2Location dataset, which has a closed methodology and unverified accuracy for the usage type attribute.

Despite our efforts, we were unable to achieve high accuracy using unsupervised and zero-shot learning methods. The primary challenge appears to stem from the complexity of hostname data, which lacks standardized naming conventions and can vary widely across network operators. The results from supervised learning were the most promising, indicating that this approach is the most effective for inferring usage types from hostname data. Certain categories, such as ISP/MOB, CDN, and MIL, were detected with relatively high accuracy, while other categories like ISP, DCH, MOB, EDU and GOV showed moderate levels of accuracy in our predictions.

For our second research question, we aimed to observe how the usage types of IP addresses evolve over time based on hostname data. Our analysis revealed several significant trends throughout 2023: the ORG category consistently declined, while CDN usage remained stable. ISP/MOB and COM both experienced steady declines over the year. DCH and ISP both showed increases over the year. SES remained relatively stable with a slight decrease towards the end of 2023. The GOV, LIB, and MIL categories exhibited little change. MOB increased significantly, especially in the later stages of the year and EDU displayed moderate but consistent growth.

Finally, to validate our methodology, we applied the supervised learning approach, using Word2Vec-extracted features that performed best for answering our first research question, to the country attribute within the IP2Location dataset. This attribute has been verified by other researchers to have high accuracy approaching 100%, and approaching that value would indicate that our methodology may work well for inferring usage types from hostnames. Our model achieved over 90% accuracy in predicting the country of IP addresses, suggesting that our methodology is robust and can be considered promising for real-world applications.

Bibliography

[1] Domain names - implementation and specification. RFC 1035, November 1987. URL: https://www.rfc-editor.org/info/rfc1035, doi:10.17487/RFC1035.

[2] 8 clustering algorithms in machine learning that all data scientists should know, n.d. Retrieved January 22, 2024. URL: https://www.freecodecamp.org/news/8-clustering-algorithms-in-machine-learning-that-all-data-scientists-should-know/.

[3] Anatomy of url, n.d. Retrieved July 26, 2024. URL: https://broken-code.medium.com/anatomy-of-url-e95ecf9bf298.

[4] Aws ip address ranges, n.d. Retrieved May 26, 2024. URL: https://docs.aws.amazon.com/vpc/latest/userguide/aws-ip-ranges.html.

[5] Bert zero-shot classification base - xlni (bert_base_cased_zero_shot_classifier_xnli), n.d. Retrieved August 27, 2024. URL: https://sparknlp.org/2023/04/05/bert_base_cased_zero_shot_classifier_xnli_en.html.

[6] A brief introduction to cluster validation, n.d. Retrieved January 26, 2024. URL: https://medium.com/@jodancker/a-brief-introduction-to-cluster-validation-ca4215295b06.

[7] Cbow — word2vec, n.d. Retrieved June 26, 2024. URL: https://medium.com/@anmoltalwar/cbow-word2vec-854a043ee8f3.

[8] Clustering algorithms, n.d. Retrieved January 22, 2024. URL: https://developers.google.com/machine-learning/clustering/clustering-algorithms.

[9] Configuring reverse dns, n.d. Retrieved January 15, 2024. URL: https://apps.db.ripe.net/docs/Database-Support/Configuring-Reverse-DNS/#reverse-dns-overview.

[10] Explore our ip address database downloads for instant access to our ip address insights, n.d. Retrieved May 26, 2024. URL: https://ipinfo.io/.

[11] Extracting feature vectors from url strings for malicious url detection, n.d. Retrieved January 17, 2024. URL: https://towardsdatascience.com/extracting-feature-vectors-from-url-strings-for-malicious-url-detection-cbafc24737a.

[12] How to beat netflix vpn ban in 2024, n.d. Retrieved January 16, 2024. URL: https://vpnpro.com/guides-and-tutorials/netflix-vpn-ban/.

[13] How to perform feature selection with numerical input data, n.d. Retrieved June 15, 2024. URL: https://machinelearningmastery.com/feature-selection-with-numerical-input-data/.

[14] Ip geolocation, n.d. Retrieved June 15, 2024. URL: https://www.ip2location.com/.

[15] Ip ranges, n.d. Retrieved May 26, 2024. URL: https://www.cloudflare.com/en-gb/ips/.

[16] maxmind, n.d. Retrieved July 15, 2024. URL: https://www.maxmind.com/.

[17] Microsoft 365 urls and ip address ranges, n.d. Retrieved May 26, 2024. URL: https://learn.microsoft.com/en-us/microsoft-365/enterprise/urls-and-ip-address-ranges?view=o365-worldwide.

[18] Pyspark overview, n.d. Retrieved July 28, 2024. URL: https://spark.apache.org/docs/latest/api/python/index.html.

[19] Reverse zones and ptr records, n.d. Retrieved January 14, 2024. URL: https://developers.cloudflare.com/dns/additional-options/reverse-zones/.

[20] Spark nlp, n.d. Retrieved July 28, 2024. URL: https://sparknlp.org/.

[21] Top netflix vpns reviewed january 2024: Unlocking content safely, n.d. Retrieved January 16, 2024. URL: https://www.vpn.com/streaming/netflix/#:~:text=in%20your%20area.-,Why%20Does%20Netflix%20Ban%20VPNs%3F,violation%20of%20their%20copyright%20agreement.

[22] Vpn datacenter ips, n.d. Retrieved May 26, 2024. URL: https://github.com/X4BNet/lists_vpn.

[23] What are mit's ip ranges?, n.d. Retrieved May 26, 2024. URL: https://kb.mit.edu/confluence/pages/viewpage.action?pageId=46301207.

[24] What is a dns zone?, n.d. Retrieved January 14, 2024. URL: https://www.cloudflare.com/learning/dns/glossary/dns-zone/.

[25] What is dns? | how dns works, n.d. Retrieved January 14, 2024. URL: https://www.cloudflare.com/en-gb/learning/dns/what-is-dns/.

[26] word2vec, n.d. Retrieved June 15, 2024. URL: https://www.tensorflow.org/text/tutorials/word2vec.

[27] Olalekan Adeyinka. Internet attack methods and internet security technology. In *2008 Second Asia International Conference on Modelling & Simulation (AMS)*, pages 77–82. IEEE, 2008.

[28] Saadaldeen Rashid Ahmed Ahmed, Israa Al Barazanchi, Zahraa A Jaaz, and Haider Rasheed Abdulshaheed. Clustering algorithms subjected to k-mean and gaussian mixture model on multidimensional data set. *Periodicals of Engineering and Natural Sciences*, 7(2):448–457, 2019.

[29] David Barr. Common dns operational and configuration errors. Technical report, 1996.

[30] Michael W Berry, Azlinah Mohamed, and Bee Wah Yap. *Supervised and unsupervised learning for data science*. Springer, 2019.

[31] Kamalpreet Bindra and Anuranjan Mishra. A detailed study of clustering algorithms. In *2017 6th international conference on reliability, infocom technologies and optimization (trends and future directions)(ICRITO)*, pages 371–376. IEEE, 2017.

[32] Barry Brown. Studying the internet experience. *HP laboratories technical report HPL*, 49, 2001.

[33] Davide Canali, Marco Cova, Giovanni Vigna, and Christopher Kruegel. Prophiler: a fast filter for the large-scale detection of malicious web pages. In *Proceedings of the 20th international conference on World wide web*, pages 197–206, 2011.

[34] Joseph Chabarek and Paul Barford. What's in a name? decoding router interface names. In *Proceedings of the 5th ACM workshop on HotPlanet*, pages 3–8, 2013.

[35] Kenneth Ward Church. Word2vec. *Natural Language Engineering*, 23(1):155–162, 2017.

[36] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany. Supervised learning. In *Machine learning techniques for multimedia: case studies on organization and retrieval*, pages 21–49. Springer, 2008.

[37] Ovidiu Dan, Vaibhav Parikh, and Brian D Davison. Ip geolocation through reverse dns. *ACM Transactions on Internet Technology (TOIT)*, 22(1):1–29, 2021.

[38] Benoit Donnet, Timur Friedman, and Mark Crovella. Improved algorithms for network topology discovery. In *Passive and Active Network Measurement: 6th International Workshop, PAM 2005, Boston, MA, USA, March 31-April 1, 2005. Proceedings 6*, pages 149–162. Springer, 2005.

[39] Ramakrishnan Durairajan, Joel Sommers, and Paul Barford. Layer 1-informed internet topology measurement. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 381–394, 2014.

[40] D Eastlake 3rd. Domain name system (dns) case insensitivity clarification. Technical report, 2006.

[41] Nadir Omer Fadl Elssied, Othman Ibrahim, and Ahmed Hamza Osman. A novel feature selection based on one-way anova f-test for e-mail spam classification. *Research Journal of Applied Sciences, Engineering and Technology*, 7(3):625–638, 2014.

[42] Paul Fergus, De-Shuang Huang, and Hani Hamdan. Prediction of intrapartum hypoxia from cardiotocography data using machine learning. In *Applied computing in medicine and health*, pages 125–146. Elsevier, 2016.

[43] Afshin Gholamy, Vladik Kreinovich, and Olga Kosheleva. Why 70/30 or 80/20 relation between training and testing sets: a pedagogical explanation. 2018, 2022.

[44] Eric L Goodman, Chase Zimmerman, and Corey Hudson. Packet2vec: Utilizing word2vec for feature extraction in packet data. *arXiv preprint arXiv:2004.14477*, 2020.

[45] Manoj Kr Gupta and Pravin Chandra. A comparative study of clustering algorithms. In *2019 6th international conference on computing for sustainable global development (INDIACom)*, pages 801–805. IEEE, 2019.

[46] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning: data mining, inference, and prediction, 2017.

[47] Trevor Hastie, Robert Tibshirani, Jerome Friedman, Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Overview of supervised learning. *The elements of statistical learning: Data mining, inference, and prediction*, pages 9–41, 2009.

[48] Steve Hawkins, David C Yen, and David C Chou. Awareness and challenges of internet security. *Information Management & Computer Security*, 8(3):131–143, 2000.

[49] Jeff Heaton. An empirical analysis of feature engineering for predictive modeling. In *SoutheastCon 2016*, pages 1–6. IEEE, 2016.

[50] Louis Hickman, Stuti Thapa, Louis Tay, Mengyang Cao, and Padmini Srinivasan. Text preprocessing for text mining in organizational research: Review and recommendations. *Organizational Research Methods*, 25(1):114–146, 2022.

[51] Bradley Huffaker, Marina Fomenkov, and KC Claffy. Drop: Dns-based router positioning. *ACM SIGCOMM Computer Communication Review*, 44(3):5–13, 2014.

[52] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[53] Jaeyeon Jung, Emil Sit, Hari Balakrishnan, and Robert Morris. Dns performance and the effectiveness of caching. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 153–167, 2001.

[54] Subbu Kannan, Vairaprakash Gurusamy, S Vijayarani, J Ilamathi, Ms Nithya, S Kannan, and V Gurusamy. Preprocessing techniques for text mining. *International Journal of Computer Science & Communication Networks*, 5(1):7–16, 2014.

[55] Hae-Young Kim. Analysis of variance (anova) comparing means of more than two groups. *Restorative dentistry & endodontics*, 39(1):74–77, 2014.

[56] Kenji Kira and Larry A Rendell. A practical approach to feature selection. In *Machine learning proceedings 1992*, pages 249–256. Elsevier, 1992.

[57] Dan Komosny, Miroslav Voznak, and Saeed Ur Rehman. Location accuracy of commercial ip address geolocation databases. *Information technology and control*, 46(3):333–344, 2017.

[58] Mikhail V Koroteev. Bert: a review of applications in natural language processing and understanding. *arXiv preprint arXiv:2103.11943*, 2021.

[59] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. Text classification algorithms: A survey. *Information*, 10(4):150, 2019.

[60] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017.

[61] Hong Liang, Xiao Sun, Yunlei Sun, and Yuan Gao. Text feature extraction based on deep learning: a review. *EURASIP journal on wireless communications and networking*, 2017:1–12, 2017.

[62] Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, and Junjie Wu. Understanding of internal clustering validation measures. In *2010 IEEE international conference on data mining*, pages 911–916. Ieee, 2010.

[63] Matthew Luckie, Bradley Huffaker, and k claffy. Learning regexes to extract router names from hostnames. In *Proceedings of the Internet Measurement Conference*, pages 337–350, 2019.

[64] Matthew Luckie, Bradley Huffaker, Alexander Marder, Zachary Bischof, Marianne Fletcher, and K Claffy. Learning to extract geographic information from internet router hostnames. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*, pages 440–453, 2021.

[65] Matthew Luckie, Alexander Marder, Marianne Fletcher, Bradley Huffaker, and K Claffy. Learning to extract and use asns in hostnames. In *Proceedings of the ACM Internet Measurement Conference*, pages 386–392, 2020.

[66] Matthew Luckie, Alexander Marder, Bradley Huffaker, and k claffy. Learning regexes to extract network names from hostnames. In *Proceedings of the 16th Asian Internet Engineering Conference*, pages 9–17, 2021.

[67] Long Ma and Yanqing Zhang. Using word2vec to process big text data. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 2895–2897. IEEE, 2015.

[68] Maher Maalouf. Logistic regression in data analysis: an overview. *International Journal of Data Analysis Techniques and Strategies*, 3(3):281–299, 2011.

[69] Hendrik Mende, Maik Frye, Paul-Alexander Vogel, Saksham Kiroriwal, Robert H Schmitt, and Thomas Bergs. On the importance of domain expertise in feature engineering for predictive product quality in production. *Procedia CIRP*, 118:1096–1101, 2023.

[70] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[71] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[72] Paul Mockapetris and Kevin J Dunlap. Development of the domain name system. In *Symposium proceedings on Communications architectures and protocols*, pages 123–133, 1988.

[73] Paul V Mockapetris. Rfc1035: Domain names-implementation and specification, 1987.

[74] Gireen Naidu, Tranos Zuva, and Elias Mmbongeni Sibanda. A review of evaluation metrics in machine learning algorithms. In *Computer Science On-line Conference*, pages 15–25. Springer, 2023.

[75] Fatemeh Nargesian, Horst Samulowitz, Udayan Khurana, Elias B Khalil, and Deepak S Turaga. Learning feature engineering for classification. In *Ijcai*, volume 17, pages 2529–2535, 2017.

[76] Nikita Patel and Saurabh Upadhyay. Study of various decision tree pruning methods with their empirical comparison in weka. *International journal of computer applications*, 60(12), 2012.

[77] Flor Miriam Plaza-del Arco, María-Teresa Martín-Valdivia, and Roman Klinger. Natural language inference prompts for zero-shot emotion classification in text across corpora. *arXiv preprint arXiv:2209.06701*, 2022.

[78] Jon Postel. Domain name system structure and delegation. Technical report, 1994.

[79] Pradeep Rai, Shubha Singh, et al. A survey of clustering techniques. *International Journal of Computer Applications*, 7(12):1–5, 2010.

[80] Steven J Rigatti. Random forest. *Journal of Insurance Medicine*, 47(1):31–39, 2017.

[81] Bernardino Romera-Paredes and Philip Torr. An embarrassingly simple approach to zero-shot learning. In *International conference on machine learning*, pages 2152–2161. PMLR, 2015.

[82] Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.

[83] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[84] Quirin Scheitle, Oliver Gasser, Patrick Sattler, and Georg Carle. Hloc: Hints-based geolocation leveraging multiple measurement frameworks. In *2017 Network Traffic Measurement and Analysis Conference (TMA)*, pages 1–9. IEEE, 2017.

[85] Yan-Yan Song and LU Ying. Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2):130, 2015.

[86] Youwei Song, Jiahai Wang, Zhiwei Liang, Zhiyue Liu, and Tao Jiang. Utilizing bert intermediate layers for aspect based sentiment analysis and natural language inference. *arXiv preprint arXiv:2002.04815*, 2020.

[87] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring isp topologies with rocketfuel. *ACM SIGCOMM Computer Communication Review*, 32(4):133–145, 2002.

[88] Lars St, Svante Wold, et al. Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4):259–272, 1989.

[89] Michael Steinbach, George Karypis, and Vipin Kumar. A comparison of document clustering techniques. 2000.

[90] Lisa Sullivan. Hypothesis testing-analysis of variance (anova). *Prospective versus Retrospective Studies*, 2016.

[91] Theresa Ullmann, Christian Hennig, and Anne-Laure Boulesteix. Validation of cluster analysis results on validation data: A systematic framework. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 12(3):e1444, 2022.

[92] Olivier van der Toorn, Roland van Rijswijk-Deij, Raffaele Sommese, Anna Sperotto, and Mattijs Jonker. Saving brian's privacy: the perils of privacy exposure through reverse dns. In *Proceedings of the 22nd ACM Internet Measurement Conference*, pages 1–13, 2022.

[93] Roland van Rijswijk-Deij, Mattijs Jonker, and Anna Sperotto. The openintel open access portal.

[94] S Vijayarani, Ms J Ilamathi, Ms Nithya, et al. Preprocessing techniques for text mining-an overview. *International Journal of Computer Science & Communication Networks*, 5(1):7–16, 2015.

[95] Ž Vujović et al. Classification model evaluation metrics. *International Journal of Advanced Computer Science and Applications*, 12(6):599–606, 2021.

[96] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265, 2018.

[97] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.

[98] Sandeep Yadav, Ashwath Kumar Krishna Reddy, AL Narasimha Reddy, and Supranamaya Ranjan. Detecting algorithmically generated malicious domain names. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 48–61, 2010.

[99] Francisco Javier Campos Zabala. Supervised and unsupervised learning. In *Grow Your Business with AI*, page Ch. 9. Apress, Berkeley, CA, 2023. URL: https://link.springer.com/chapter/10.1007/978-1-4842-9669-1_9, doi:10.1007/978-1-4842-9669-1_9.

[100] Li Zhang, Jun Li, and Chao Wang. Automatic synonym extraction using word2vec and spectral clustering. In *2017 36th Chinese Control Conference (CCC)*, pages 5629–5632. IEEE, 2017.