# POLITECNICO DI TORINO

## MASTER's Degree in CYBERSECURITY


## MASTER's Degree Thesis

# Applying Neural Topic Modeling to Detect Anomalous Permission Requests in Microsoft 365 Applications

**Supervisors**

**Prof. Marco MELLIA**

**Nikhil JHA, Ph.D.**

**Dr. Alberto VERNA**

**Candidate**

**Vincenzo LONGO**

**OCTOBER 2025**

# Applying Neural Topic Modeling to Detect Anomalous Permission Requests in Microsoft 365 Applications

**Vincenzo Longo**

## Abstract

Microsoft 365 is a comprehensive suite of productivity tools that offers users a variety of services over the cloud. A significant advantage of Microsoft 365 is its ability to generate private cloud environments for organizations, called tenants or directories, where users have their own identities, can access their data, and collaborate with each other in a controlled environment. Applications are a key component for improving productivity and enhancing collaboration within tenants. They can request a set of permissions theoretically required to perform their intended functionalities. These permissions allow the applications to get read or write access to various tenant's resources such as user information, calendars, files, mailbox, and more. Consequently, the misuse of these permissions may represent a potential entry point for attackers who may compromise an existing application with excessive permissions or develop a malicious application to access sensitive data or perform unauthorized actions.

This research aims to intersect the applications' intents with the permissions they request to identify potentially anomalous applications. For instance, a calendar manager application requesting permissions to access users' files or mailboxes could be considered anomalous. We first put significant efforts into collecting a dataset of applications with required permissions and their descriptions. Indeed, a major challenge in this context was the limited availability of data and labelled samples due to the restricted and fragmented nature of Microsoft 365 environments. Once the dataset was collected, we leveraged Neural Topic Modelling techniques to extract the intents of applications from their descriptions and clustering them based on the identified topics. At the end, a mixed unsupervised-supervised approach is proposed to build a model capable of identifying applications with potentially anomalous permission requests within the same topic.

# Dedications

*to my grandparents, in their loving memory*

ACKNOWLEDGMENTS

Prima di tutto, desidero esprimere un sincero ringraziamento ai miei relatori, Prof. Marco Mellia, Nikhil Jha e Alberto Verna per la loro guida, supporto e pazienza durante il mio percorso di tesi. I loro consigli e suggerimenti sono stati fondamentali per il successo di questo lavoro.

Un ringraziamento speciale va ai miei genitori, a papà Salvatore e mamma Carmela, per il loro amore incondizionato e il sostegno costante, e a mia sorella Maria Giovanna per essere sempre al mio fianco. Senza di loro, non sarei arrivato fin qui. Un grazie di cuore anche a tutta la mia famiglia, in particolare ai miei zii e mio cugino Vincenzo, per il loro affetto e incoraggiamento. Desidero inoltre ringraziare gli amici universitari che mi hanno e continuano ad accompagnare in questa parte della mia vita. Infine, un ringraziamento speciale agli amici di sempre che in ogni occasione trovano il modo di farmi sorridere e di starmi vicino. Grazie di cuore a tutti voi.

# Table of Contents

# List of Figures

# List of Tables

# Acronyms

| | |
|---|---|
| M365 | Microsoft 365. |
| IAM | Identity and Access Management. |
| CRM | Customer Relationship Management. |
| MFA | Multi-Factor Authentication. |
| OAuth | Open Authorization. |
| RSC | Resource-Specific Consent. |
| MDA | Microsoft Defender for Cloud Apps. |
| API | Application Programming Interface. |
| SIEM | Security Information and Event Management. |
| DLO | Data Loss Prevention. |
| NLP | Natural Language Processing. |
| NTM | Neural Topic Modelling. |
| LDA | Latent Dirichlet Allocation. |
| FNR | False Negative Rate. |
| TPR | True Positive Rate. |
| NPMI | Normalized Pointwise Mutual Information. |

# Chapter 1

# Introduction

## 1.1 Context

The shift to cloud computing and the advent of cloud-based productivity suites have represented a revolution in the way organizations manage their IT resources, enabling companies to access their data from anywhere at anytime. Any type of organization, from small businesses to large enterprises, can benefit from the flexibility, scalability, and ease of use that cloud services offer. Cloud computing allows organizations to access, store and manage resources without the need of handling physical IT infrastructures, cutting costs and improving efficiency. Today, cloud-based productivity suites are widely adopted by organizations. The question is no longer to migrate to the cloud, but rather what to migrate and which cloud provider to choose [1]. Microsoft and Google are the two main players in this market offering their respective suites Microsoft 365 (M365) and Google Workspace.

M365 is the cloud-based solution for productivity developed by Microsoft. It occupies 30% of the global enterprise IT market with more than one million companies from around the world using its services [2]. Its offer includes the possibility of having online access to the most recent versions of Microsoft Office applications such as Word, Excel, PowerPoint and Outlook, along with advanced collaboration tools such as Microsoft Teams for chat and video conferencing, SharePoint Online for document management, Exchange Online for email and calendaring, OneDrive for data storage, and many other services [3]. These solutions significantly enhance organizational productivity, offering a set of cloud-based services providing benefits like enterprise data security, daily productivity support, real-time collaboration and communication [4, 5, 6].

This complete suite of services is provided to each organization within a secure and isolated environment known as *tenant*. Each tenant is a dedicated and unique instance of Microsoft 365 services and organizational data, completely separate from all other tenants [7]. It is the logical representation of the organization where all the enterprise's entities (users, devices, applications) and data are securely preserved, made accessible when needed, and protected from unauthorized access.

The level of productivity and collaboration offered by M365 can be further

extended by integrating applications that can directly handle vital tasks during users daily workflow. The functionalities offered by these applications range from simple tasks, such as sending notifications or reminders, to more complex operations, such as project management, customer relationship management (CRM), communication tools, etc. These can be first-party applications developed by Microsoft, or third-party applications, developed by external vendors that integrate with Microsoft 365 built-in services. For instance, a user within an organization can integrate Word with an application like Wikipedia to quickly search for information, Teams with Zoom to schedule and join video meetings and Youtube to watch videos during a call, or Outlook with Trello to manage tasks and projects.

In this vast ecosystem, applications are not just code running on user's devices, but they are first-class citizens of the tenant. When a user installs an application, it is registered and entered the tenant as any other user do. This means that each application has its own identity and privileges, and it can be authenticated and authorized enabling for more secure and flexible interactions. Indeed, applications can request permissions, which define the tasks that the application can perform and the resources it can access within the tenant. Any time a user adds an application to his M365 environment, he is asked to consent a set of permissions, that if granted, allow the application to access specific resources. The scope of a permission widely varies, as it can range from just reading user profile information to full control over user's or organizations's data like emails, files, calendars, etc.

## 1.2 Motivations and Methodology

The rise of cloud environments in the last decade shows at the same time the increasing of the number of attacks targeting organizations over the cloud, with threat actors mainly exploiting misconfigurations and human errors [8]. Forbes defined applications as "an overlooked vulnerability" [9], since users often can install applications without any control and these applications can request excessive permissions either if the app was legitimate or malicious. Indeed, applications and their permissions represent a critical vulnerability that if exploited can grant attackers wide access to organizational resources even though strong authentication mechanisms like Multi-Factor Authentication (MFA) are in place [10]. The risk becomes even more relevant when considering that typically developers prioritize functionality over security, without taking care about the principle of least privilege [11]. This principle involves granting any entity only those privileges that are essential to perform its intended functionalities [12]. For instance, an application for the only purpose of managing user's meetings should not request permissions to read or write user's files or emails.

For this reason, understanding the intents of an application is crucial before reviewing the permissions it requests. This thesis addresses this issue by proposing a methodology to find discrepancies between the permissions requested and the intents of the application.

The main research question is

*"Is it possible to automatically identify overprivileged applications in the Microsoft 365 ecosystem?"*

To achieve this goal, it is necessary to automatically extract the declared intents of each application and compare them with the permissions it requests. By doing so, it is possible to identify applications whose requested permissions exceed what is necessary for their intended functionality, highlighting potentially overprivileged applications that could pose a threat to the security of an organization.

This is achieved by applying a *Neural Topic Modelling* (NTM) technique on the applications descriptions to extract the main topics populating the Microsoft 365 applications domain, and a mixed unsupervised-supervised approach to identify overprivileged applications in each cluster. We collected application data from multiple sources utilizing the Politecnico di Torino applications dataset (provided by the Microsoft IT department of the university), official Microsoft APIs and web scraping techniques. At the end we obtain a dataset of about 1,000 applications all with their description and set of permissions. The results will show that the proposed approach is promising in detecting overprivileged applications that would be otherwise missed without considering a semantic context.

This work aims to provide a first step towards a better understanding of the Microsoft 365 applications ecosystem and to raise the awareness about the risks associated to over privileged applications.

## 1.3 Thesis Structure

The thesis is structured as follows.

- Chapter 2 provides the necessary background knowledge about the Microsoft 365 ecosystem, its permission model, related security risks and the existing solutions to mitigate them.
- Chapter 3 presents an overview of the related works in the field of NTM and anomaly detection.
- Chapter 4 describes the data collection process and the datasets obtained.
- Chapter 5 illustrates the proposed methodology to extract topics and identify overprivileged applications.
- Chapter 6 discusses the results obtained from the experiments.
- Chapter 7 concludes the thesis and suggests future research directions.

# Chapter 2

# Microsoft 365 Ecosystem

In this chapter, we provide an overview of the Microsoft 365 ecosystem, what it offers and how it is structured. We also introduce the concept of applications within a tenant and the permissions model that governs the access to resources. Finally, we introduce the least privilege principle and the risks coming from its poor implementation. Some examples of attacks exploiting over-privileged applications are provided, along with the existing mitigation strategies offered by Microsoft.

## 2.1 Microsoft 365

M365 is a cloud-based productivity suite that helps organizations and individuals to collaborate with each other enhancing their productivity, being a comprehensive solution for modern workplaces. It represents a transition from traditional software package locally installed on users devices to a cloud-based subscription service enabling users to access applications and services from any device with an internet connection. By subscribing to Microsoft 365, users always have access to the latest versions of the M365 productivity applications and services [4, 13]. M365 was formally known as Office 365 and included traditional Office applications (Word, Excel, PowerPoint, etc.) and the first cloud services for mail (Exchange Online) and data storage (SharePoint Online). Later, Microsoft renamed most of the versions of Office 365 to Microsoft 365 reflecting a significant evolution towards a more comprehensive, integrated platform. The major additions include enterprise-grade security tools, device management services, AI features, offering a single "productivity cloud" for modern work [14]. Today, the M365 platform offers a wide range of tools categorized by their functionalities [4, 15]:

- **Productivity:** Word, Excel, PowerPoint, OneNote.
- **Communication:** Teams, Outlook, Exchange Online.
- **Data storage:** OneDrive, SharePoint.
- **Integrated Security:** Microsoft Defender, Microsoft Purview.

All of these services, along with the users and the applications that interact with them, are managed within a dedicated environment called *tenant*.

### 2.1.1 Microsoft Entra ID

M365 tenants are managed by Microsoft Entra ID (formerly Azure Active Directory). This is the cloud-based Identity and Access Management (IAM) solution of Microsoft that provides the essential identity, authentication, policy, and protection to secure users, devices, apps, and resources [16]. Specifically, each Microsoft Entra ID tenant has its dedicated and trusted Entra ID directory (database) which includes all the *security principals* [16, 17, 7]. A security principal represents any user, group, or application which can be authenticated and authorized to access resources within the Entra ID tenant. The main types of security principals in a Entra ID directory are [18, 19, 20, 21]:

- **Users**, representing individuals within the tenant.
- **Groups**, representing collections of users or other groups that can reduce manual tasks and simplify authorization management.
- **Applications**, representing client applications or services that can be registered in the Entra ID tenant to access resources.
- **Devices**, representing the identity of devices registered in the Entra ID tenant.

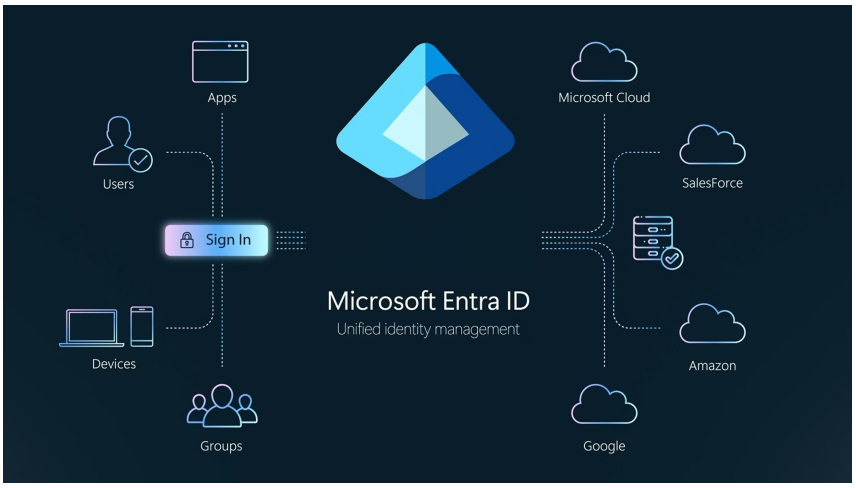


**Figure 2.1:** Microsoft Entra ID tenant structure and main entities. Source: Microsoft.

### 2.1.2 Authentication and Authorization in M365

The two fundamental features of an identity platform like Entra ID are authentication and authorization. Authentication is the process of verifying an identiy claim (proving if an entity is who it claims to be), while authorization is the process of determining the access rights of a verified identity (deciding what it is allowed to do). Therefore,

authentication is the prerequisite of authorization [22, 23, 24]. Microsoft Entra ID primarily uses the OAuth 2.0 protocol for authorization, which is the industry-standard protocol. The OAuth 2.0 authorization flow involves some key actors: the client requesting access to a protected resource, the resource owner which possesses the resource, and the authorization server which issues *access tokens* to the client with the approval of the resource owner. Once the client has obtained the access token, it can use it to access the resource from the server hosting it [25].

## 2.2 Applications in M365

Applications in Microsoft Entra ID represent a client application or service with their own identity configuration registered in the Entra ID tenant. At registration time, the developer of the application must provide the necessary information and configuration settings to define the applications such as the redirect URIs and the supported multi-tenant access. Once the application is registered, a globally unique instance of the application is created, known as the *application object*. This application object defines the application's identity configuration and it is uniquely identified by an application (client) ID. A corresponding *service principal* object is created in the tenant when the application is assigned to at least one user in the tenant. The service principal is the operational representation of the application within a specific tenant. Practically, an application object is used as a template to create one or multiple service principal objects in the current and other tenants. The diagram in Figure 2.2 better depicts the relationship between application objects and service principal objects in the context of a sample multitenant application called HR app. There are three Microsoft Entra tenants in this example [26]:

- Adatum - The producer tenant, used by the company that developed the HR app.
- Contoso - The first consumer tenant, used by the Contoso organization, which uses the HR app.
- Fabrikam - The second consumer tenant, used by the Fabrikam organization, which also uses the HR app.

The application object of the HR app only resides in the producer tenant (Adatum) and it is used as a baseline to create all the service principal objects that would be created in any tenant using the app (like Contoso and Fabrikam).

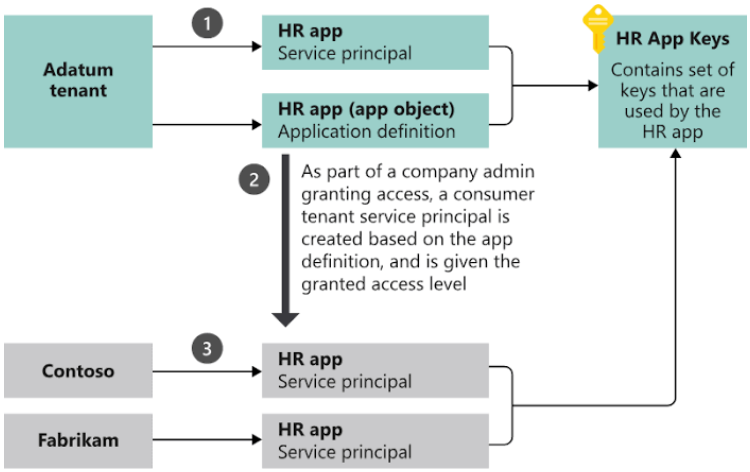


**Figure 2.2:** Relationship between application object and service principal object. Source: Microsoft.

## 2.3 OAuth Permission Model in M365

The OAuth 2.0 framework manages authorization by means of access tokens, which contain specific permissions. Access tokens are strings which represent the authorization issued to the client by the authorization server. It represents the key used to access the protected resource. Access tokens include permissions which directly define the set of capabilities provided by the token. OAuth defines permissions as a mechanism to limit the application's access to resources. The access token is intended to be read and used by the resource server to determine whether the client actually has the right to access the requested resource and which is its level of access [27].

### 2.3.1 Permissions Types

M365 has two main types of OAuth permissions [11, 28]:

- **Delegated permissions**: used by applications that act on behalf of a signed-in user. In this case, the scope of the application's access is restricted to the user's resources. Indeed, these permissions are also known as "scopes" and they are determined by the intersection of privileges granted to the application and thos granted to the signed-in user.
- **Application permissions:** used by applications that run without a signed-in user granting tenant-level access to the resource specified in the permission. These permissions are also known as "app-roles", and they are typical of background services or daemons. Application permissions can be granted only by global administrators of the tenant, as they grant tenant-wide access to the organization's resources.

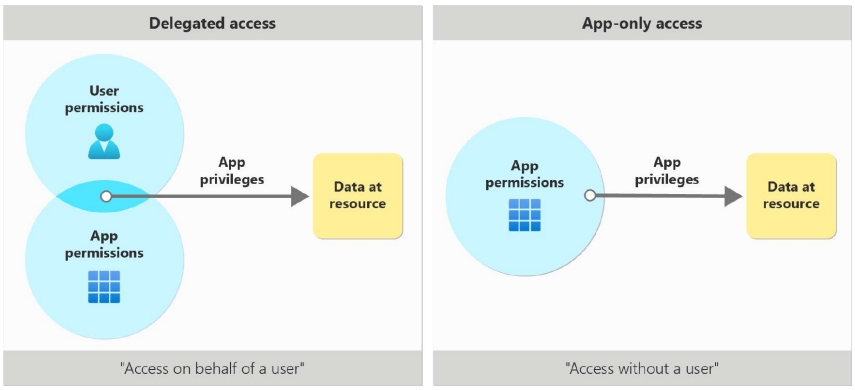


**Figure 2.3:** Delegated and Application permissions in M365. Source: Microsoft.

### 2.3.2 Resource-Specific Consent in Teams Applications

In addition to the standard permission model, Microsoft introduced **Resource-Specific Consent** (RSC) [29], which is an authorization model implemented by Microsoft Teams which enables scoped access to applications. RSC allows authorized users to grant an application scoped access to the data of a specific team or chat, instead of giving access to all the user's data. This model is particularly useful because it allows for fine-grained access control, avoiding the need to grant broad tenant-level permissions. For instance, a person owning both team A and team B can consent to an application to access only the resources of team A without affecting team B.

### 2.3.3 API Permissions and Microsoft Graph

An application, acting as the client, must interact with a resource target application that exposes the API to access its resources. The primary API for accessing M365 services and data within a Microsoft Entra ID tenant is the Microsoft Graph API which can be used to build applications for organizations and consumers operating in the M365 ecosystem. The Graph API was designed to unify existing legacy APIs such as the Office 365 SharePoint Online API, the Office 365 Exchange Online API, etc. [28].

### 2.3.4 Permissions Pattern

The API of the resource server uses permissions to restrict access to its resources. OAuth standards do not define a specific pattern for permissions naming, but leave it to the developer of the API. However, a best practice is to use the following naming pattern [30, 31].

`<resource>.<operation>.<constraint>`

This pattern is highly expressive and allows to easily capture the intent of the permission. Each field of the pattern is described in Table 2.1.

| Field | Description | Most Common Examples |
|---|---|---|
| `resource` | Refers to a Microsoft Graph resource to which the permission allows access. For example, the user resource. | User, Chat, Mail, Files, ... |
| `operation` | Refers to the Microsoft Graph API operations that are allowed on the data exposed by the resource. | Read, ReadWrite, ReadBasic, Create, Send, ... |
| `constraint` | Determines the potential extent of access an app has within the directory. This value may not be explicitly declared. When undeclared, the default constraint is limited to data owned by the signed-in user. | All, Directory, OwnedBy (DEFAULT), Shared, ... |

**Table 2.1:** Microsoft Graph API permission naming pattern fields

For instance, the `User.Read` permission means that the application has access to the `User` resources, with the `Read` operation (access level). Additionally, the access to the resources is limited by the `OwnedBy` constraint (default value) which means that the application can only access the profile of the signed-in user. On the other hand, the `User.Read.All` permission means that the application can read (`Read`) the profile of any user (`User`) in the directory (`All` constraint).

### 2.3.5 The Authorization Code Flow

The OAuth 2.0 authorization code flow is a secure, multi-step process, that allows a client application to obtain an access token. The high-level steps of the authorization code flow are the following [32]:

1. The user accesses the client application and initiates the sign-in process.
2. The application redirects the user to the Microsoft OAuth 2.0 authorization endpoint (`login.microsoftonline.com/common/oauth2/v2.0/`).
3. The user selects the account and completes the authentication process.
4. The user is prompted to consent to the permissions requested by the application.
5. After giving consent, the Microsoft authorization endpoint redirects the user back to the application with an authorization code, which is a temporary code that the application should use to request the final access token.

6. The application uses the authorization code to request the access token from the Microsoft token endpoint.

7. At this point, the application has the "key" (the access token) enabling it to call the API of the resource server to perform the requested and authorized operations.

## 2.4 The Principle of Least Privilege in Modern Security

The principle of least privilege (PoLP) is a fundamental pillar in modern cybersecurity, designed to limit potential damage from security incidents [33]. The U.S. National Institute of Standards and Technology (NIST) defines it as follows [34]:

> "A security principle that a system should restrict the access privileges of users (or processes acting on behalf of users) to the minimum necessary to accomplish assigned tasks."

Today, this principle represents an important cybersecurity construct, especially for organizations operating within a cloud environment such as a Microsoft Entra ID tenant, where not only users but also applications can access and manage critical resources and data. In such a context, PoLP helps to minimize the attack surface and protect the organization from the financial, data and reputational losses that result from cyberattacks and human errors [33].

### 2.4.1 Applying PoLP to M365 Applications

As discussed in Section 2.2, an application is a security principal that can be granted permissions to access resources within the directory. It can act on behalf of a user (delegated permissions) or independently (application permissions) and it can directly perform sensitive tasks such as reading mails, sending messages, or accessing files. Therefore, the PoLP must be applied not only to users but also to applications. When a developer designs an application, it should request only the minimum set of permissions necessary for the application's functionalities. This requires a careful evaluation of which API calls a permission enables, and whether it is strictly necessary or not.

Microsoft Graph provides a wide range of fine-grained permissions that are perfectly suited to implement the PoLP. For instance, apps that only need to read user profiles and send messages to user's chats should request only the `User.Read` and `ChatMessage.Send.Chat` delegated permissions instead of the more powerful `User.ReadWrite` and `Chat.ReadWrite` permissions which would allow the app to modify the user profile and write all users chats [35]. However, the enforcement of this principle strictly relies on the consent process. When users add applications to their tenant, they are presented with a list of required permissions and must evaluate whether to grant access. This represents a significant challenge, as users typically do not have the technical expertise to judge whether a requested permission is truly

justified for the application's stated purpose. This gap between the ideal application of PoLP by developers and its practical enforcement by users is a key area of security risks.

### 2.4.2 The Permission Gap

A *permission gap* arises when an application request permissions that remains unused or can be reduced with no impact over the core functionalities of the application. This disconnection comes from various factors:

- **Over-privileged applications**: Any application with unused or reducible permissions due to design flaws.
- **Malicious applications**: Some applications might be designed with malicious intents, requesting excessive permissions that could be exploited for unauthorized access or data exfiltration.

For instance, consider the case of an application named "MyCalendarApp" with the following description:

> "MyCalendarApp is the application that helps you to manage your calendar events, schedule meetings suggesting participants efficiently from your contacts."

However, when users add this application to their tenant, they are requested to consent to the following permissions:

- `Calendars.ReadWrite` - Read and write access to user's calendars.
- `Contacts.Read` - Read access to user's contacts.
- `User.Read` - Read access to user's profile.
- `Files.ReadWrite.All` - Read and write access to all user's files.

Here the permission-intent disconnection is evident. The first three permissions are necessary for the application's core functionality, but the `Files.ReadWrite.All` permission is excessive and not justified by the application's purpose. If misused, this permission could allow the application to access and modify all user's files, leading to potential ransomware attacks, or data exfiltration.

On the other hand, the case of reducible permissions is more difficult to detect. Let's assume now that "MyCalendarApp" wanted to implement a feature that allows the app to read user's files and provide a summary about the projects he's working on. In this case the access to file is justified, but the `Files.ReadWrite.All` permission is still excessive and can be reduced by replacing it with the `Files.Read` permission which limits the privileges of the application to only reading user's files [36].

This disconnection is a strong indicator of either a potentially malicious application or a dangerously over-privileged one (poorly designed). Addressing this disconnection

is crucial for enhancing security in the M365 ecosystem, as it helps to ensure that applications operate within the boundaries of their intended functionalities, reducing the surface area for potential attacks.

## 2.5 Threat Models and Existing Mitigation Strategies

The permission gap introduced in Section 2.4.2 is not just a theoretical vulnerability, but one that is practically exploited by attackers. This section introduces the main threat models that exploit this vulnerability and the existing mitigation strategies provided by Microsoft to face them.

### 2.5.1 Threat Models

There are two main ways to exploit the permission gap: the **illicit consent grant** and the **abuse of over-privileged applications**.

#### Illicit Consent Grant

This is the case of a malicious application that tricks users into granting over-privileged permissions. The attack is built around social engineering techniques and follows these steps [37].

1. The attacker registers a multi-tenant application in his own Entra ID tenant, setting the permissions he wants to request (typically needed for further attacks).
2. The attacker sends the authorization link to a user of the target tenant, using social engineering techniques to convince him to proceed with the authorization.
3. The user gives consent to the application, granting it the requested permissions.
4. The attacker retrieves the authorization code.
5. The attacker exchanges the authorization code for the access token.
6. Now the attacker is within the target tenant and can use the access token to call the APIs it has permissions for.

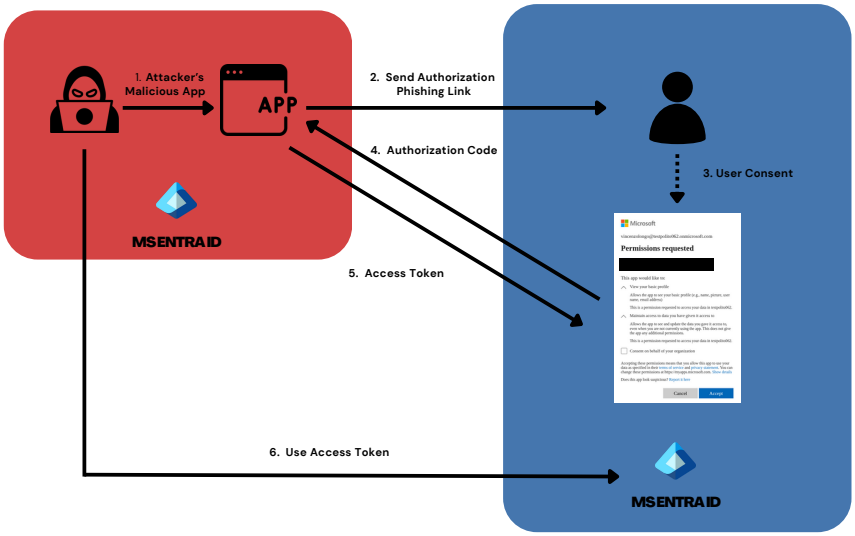


**Figure 2.4:** Illicit Consent Grant attack flow.

### Abuse of Over-Privileged Applications

Another threat model that exploits the permission gap is the abuse of over-privileged applications. This is the case of applications that did not respect the PoLP, being a potential backdoor for attackers. There are two main scenarios related to this threat model [36]:

1. the attacker directly compromises the over-privileged application, exploiting the high level of access it has within the tenant,

2. the attacker compromises the account of a developer who owns one or more applications active within the tenant. Then, the attacker can access to the tokens of the applicatiions owned by that developer and use them to access the resources of the tenant.

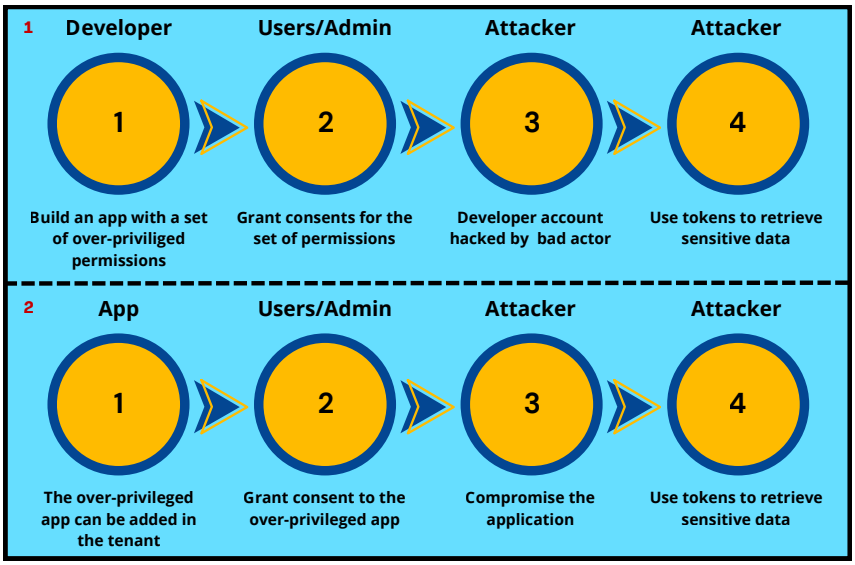


**Figure 2.5:** Abuse of Over-Privileged Applications attack flow. Source: Microsoft.

### 2.5.2 Existing Mitigation Strategies

Microsoft offers a wide range of built-in security tools and best practices to help organizations protect their M365 tenants. In this section, we provide a brief overview of the tools and the practices that aim to mitigate the risk related to the permission gap [38, 39]:

#### Admin Consent Workflow

This is a best practice that admins should implement to prevent users from granting consent to all permissions requested by applications. If this option is enabled, users do not have the ability to directly consent to applications' permissions. Instead, they can send to the global administrator a request to approve the application. In this case, the admin acts as a reviewer which should have the proper technical knowledge to judge the correctness of the requested permissions. However, reviewing permissions manually is a time-consuming and difficult task, even for experts [38].

#### Microsoft Defender for Cloud Apps

Microsoft Defender for Cloud Apps (MDA) is the Microsoft solution for protecting cloud applications. It provides several functionalities, including Cloud Access Security Broker (CASB) capabilities, Security Posture Management (SPM), and advanced threat protection. Furthermore, it offers control over OAuth applications via the App Governance, which is an MDA add-on, that helps for watching for unused applications, monitoring expired credentials and governing the apps within the tenant [39].

App Governance threat detection is based on log analysis and a set of built-in policies and provides security alerts for suspicious application activities. The data used for the detection is collected from different sources, including Microsoft Entra ID logs, Cloud App Security logs, and other Microsoft 365 services. The App Governance collect information and provides a report in a single panel accessible from the compliance center (compliance.microsoft.com) [40].

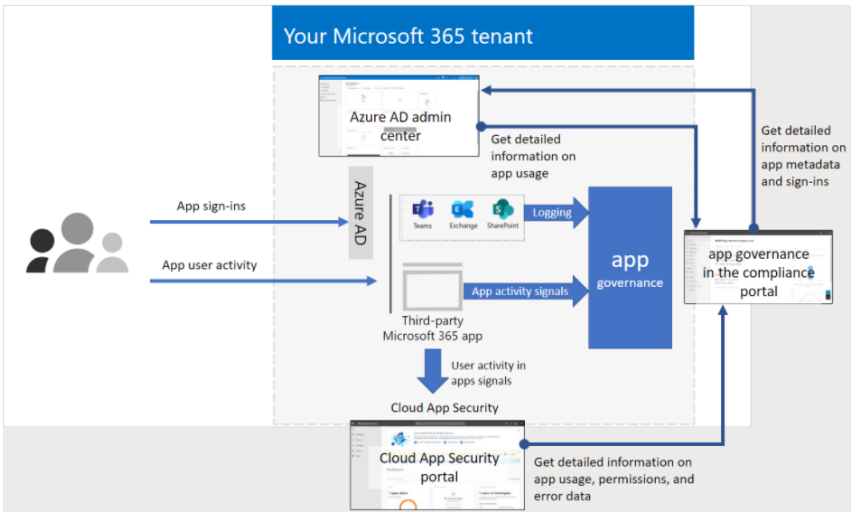


**Figure 2.6:** Microsoft Defender for Cloud Apps App Governance. Source: Cloud Architekt.

Each alert produced by App Governance is associated with by a MITRE ATT&CK tactic. Here some examples of alerts related to the permission gap [41] [1]:

- **Collection**: an app made unusual search activities in the organization. This alert is triggered when a suspicious application performs a high volume of search activities (e.g., for specific content in emails or files) by using the Microsoft Graph API.
- **Initial Access**: an app sets as redirect URL a phishing domain. This exploits the OAuth redirection vulnerability.
- **Persistence**: an app made anomalous API calls to update or add new credentials.

MDA offers the possibility to ban applications that are not aligned with the organization's security policies. If the application is banned, the access to the app is blocked from Microsoft Entra ID and the app cannot access any resource within the tenant [40].

[1] The complete list of alerts is available at `https://learn.microsoft.com/en-us/defender-cloud-apps/app-governance-anomaly-detection-alerts`.

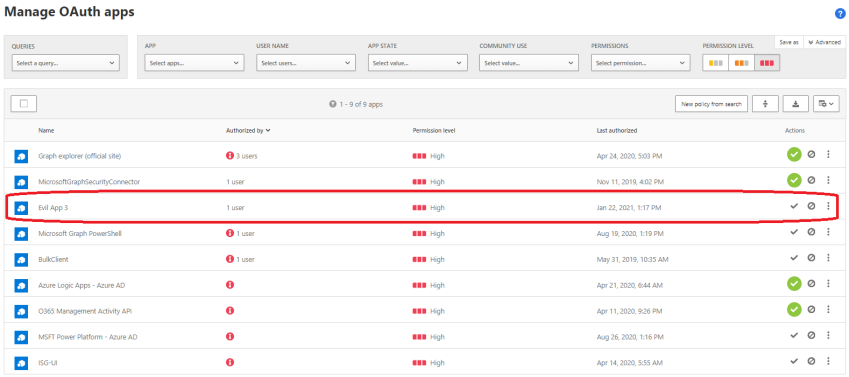


**Figure 2.7:** Microsoft Defender for Cloud Apps App Governance dashboard. Source: Cloud Architekt.

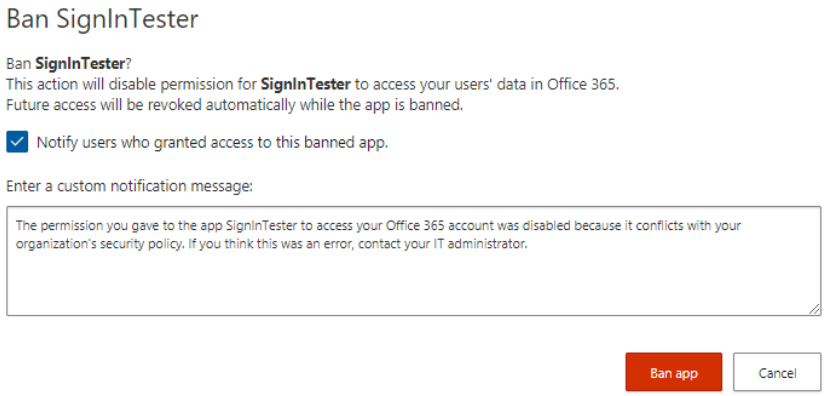


**Figure 2.8:** Ban Applications in MDA. Source: Cloud Architekt.

### Other Built-in Security Tools

Additionally, Microsoft provides other built-in services for detecting any attack that could be consequent to the exploitation of the permission gap. These reactive tools are the following [42, 43]:

- **Microsoft Sentinel**: it is the Microsof Security Information and Event Management (SIEM) solution, integrating log analytics, threat intelligence and AI to provide complete reviews of the tenant's security posture.
- **Microsoft Purview**: it is the Microsoft Solution for Data Loss Prevention (DLP), which helps protecting from data leakage and over-sharing of information.

It is evident that these strategies are strongly reactive, as they focus on detecting and sending alerts. This work proposes a proactive strategy based on machine learning to automatically detect over-privileged permissions by comparing the application permissions with its stated intents, to identify risks before the application is added to the tenant.

# Chapter 3

# Related Work

This chapter introduces the key works that inspired this thesis. Firstly, we introduce studies centered around the Microsoft 365 ecosystem and the detection of attacks like illicit consent grants. We present the main studies in this field, highlighting the limitations of rule-based systems and the advantages that machine learning techniques can provide.

Subsequently, we discuss the improvements in the field of malware detection leveraging Natural Language Processing (NLP) techniques to capture the semantic meaning of the applications. We present the state-of-the-art model in this field, which is the basis of our work. These latter works are all related to Android malware detection, as the huge availability of labelled datasets made this platform the most suitable for research. This review is meant to show the effectiveness of these techniques in the field of malware detection and highlight the applicability of similar techniques for our work.

## 3.1 Detecting Illicit Consent Grant in Microsoft Entra ID

This section presents the main works [44, 45] involving the detection of illicit consent grants in Microsoft Entra ID. The first work is from a public repository on GitHub which provides a collection of playbooks for the detection of attacks in Microsoft Entra ID including illicit consent grants. These solutions are all based on detection rules which play a significant role in detection. The second work is a thesis from Zannone et al. [45] which shows how machine-learning techniques are better at detecting attack patterns with respect to rule-based systems.

The authors of [44] simulate several attack scenarios in Microsoft Entra ID, including the Illicit Consent Grant attack, to evaluate the effectiveness of various rule-based approaches. The playbooks use detection rules designed by experts in Azure Active Directory (AAD) security, leveraging not only sign-in and audit logs but also data from other Microsoft and third-party solutions. Additionally, Sentinel analytic rules are explored, which rely only on AAD audit and sign-in logs. The rules result easy to be customized and integrated within a tenant, but, as with all

heuristic or signature-based approaches, they may fail as they cannot detect novel or sophisticated attacks bypassing predefined patterns.

Zannone et al. [45] propose a machine-learning-based approach to detect attacks in Microsoft Entra ID, including Illicit Consent Grants. The authors compared the performance of various state-of-the-art machine learning-based anomaly detection techniques with the traditional rule-based approaches. They selected two anomaly detection algorithms, Isolation Forest (iForest) and Autoencoder (AE), and applied them to a mix of real-world and simulated attack data. The feature engineering process was crucial in this work, as the authors created and extracted relevant features from the logs to help the algorithms in detecting anomalies. The evaluation results demonstrated that machine-learning techniques outperformed the traditional rule-based techniques, achieving a low false positive rate while being effective in detecting sophisticated attack scenarios.

## 3.2 Malware Detection and Topic Modelling in Malware Detection

Installing an application always comes with the risk that it might be malicious. For this reason, techniques such as static and dynamic code analysis have been widely used to detect malware. Early works show that machine learning techniques have been widely used in behaviour-based detection [46]. For instance, DREBIN [47] performs static analysis of Android applications, extracting as many features as possible from the application code and manifest such as permissions, API calls and network addresses.

However, both these techniques have limitations, as static analysis can be easily evaded by employing techniques such as code obfuscation, and dynamic analysis is resource consuming. Hence, researchers started to explore the possibility of using easily collectable metadata features to build stronger context and improve the detection. This is enforced by the fact that anomalous behaviour always depends on the context, and this is particularly true in the field of application security. For instance, a feature that is anomalous for one application can be perfectly normal for another one [48]. Late improvements in NLP, and the introduction of Neural Topic Modelling (NTM) Techniques such as Latent Dirichlet Allocation (LDA) [49] and BERTopic [50], inspired researchers to extract semantic features to intersect context and behaviour.

A first attempt to use NTM techniques in malware detection was made by Gorla et al. [48], who proposed CHABADA, a system that leveraging Latent Dirichlet Allocation (LDA) plus K-Means to extract topics from the application descriptions, was able to flag novel malwares without adding any known malware pattern.

Further improvements were made by Ranaweera et al. [51], who proposed BERT-Detect, a system that leverages BERTopic to extract topics achieving improvements in False Negative Rate (FNR) and True Positive Rate (TPR).

In the next section we present in detail the BERTDetect system and the BERTopic models, as they are the basis of our work.

## 3.3 BERTopic and the BERTDetect Framework

Topic Modelling is a technique that allows to extract topics from collections of documents with the goal of extracting their latent semantic structure in a completely unsupervised fashion. The rise of NLP has facilitated the development of more sophisticated Neural Topic Models [52].

Among these, BERTopic [50] is one of the most recent and effective models for topic extraction, as it leverages the power of transformer-based embeddings, unlike probabilistic bag of word models such as LDA [53]. BERTopic is proved to be effective in capturing semantic similarity among documents, remaining competitive across a wide range of benchmarks involving traditional models in topic modeling. Additionally its modularity makes it adaptable to evolutions in the field of NLP (i.e, new and more effective language models can be easily integrated in the pipeline) and easy to be fine-tuned for specific tasks.

BERTopic is the core of BERTDetect [51], a framework for Android malware detection, that leverages Neural Topic Modelling and API calls to improve the detection of novel malwares. The main intuition behind BERTDetect is that applications with same intents are likely to have same behaviour. The framework is composed by two main steps:

1. **Topic Extraction**: in this step, the descriptions of applications are clustered using BERTopic to capture semantic similarity,
2. **Anomaly Detection**: in this step, OC-SVM is used for malware detection in each topic.

The complete pipeline is shown in Figure 3.1. Specifically, the model is trained on a dataset of 5000 benign applications. This means that both the BERTopic model for topic extraction and the per-topic OC-SVM models for anomaly detection are trained only on benign applications. At inference time, the application description is taken as input, the highest probable topic is extracted and then the corresponding OC-SVM model is used to classify the application as benign or malicious. The framework was evaluated with three baselines:

- **LDA + OC-SVM**: LDA is used for topic extraction and OC-SVM for anomaly detection,
- **CHABADA**: LDA + K-Means is used for topic extraction and OC-SVM for anomaly detection,
- **C-GATA**: it uses GPT-based CATegorization of Android apps (C-GATA) [54] for topic extraction and OC-SVM for anomaly detection.

| Method | TN Rate | FP Rate | FN Rate | TP Rate | F1 Score |
|---|---|---|---|---|---|
| LDA | 85.60% | 14.40% | 80.80% | 19.20% | 0.25 |
| CHABADA | 83.80% | 16.20% | 68.75% | 31.25% | 0.37 |
| G-CATA | **86.60%** | **13.40%** | 57.14% | 42.86% | 0.49 |
| BERTDetect | 82.40% | 17.60% | **49.11%** | **50.89%** | **0.54** |

**Table 3.1:** Baselines vs BERTDetect - Performance Comparison [51]. Bold values indicate the best performance for each metric.

The results (reported in Table 3.1) show that BERTDetect provides an overall improvement of F1 score. G-CATA manages to achieve a good TNR showing that transformer based models are more effective in capturing the semantic similarity among descriptions. However, BERTDetect shows a significant improvement in FNR and overall F1 score. Other baselines do not manage to achieve these results, mainly due to the limitations of LDA in capturing semantic similarity with respect to transformer-based models.

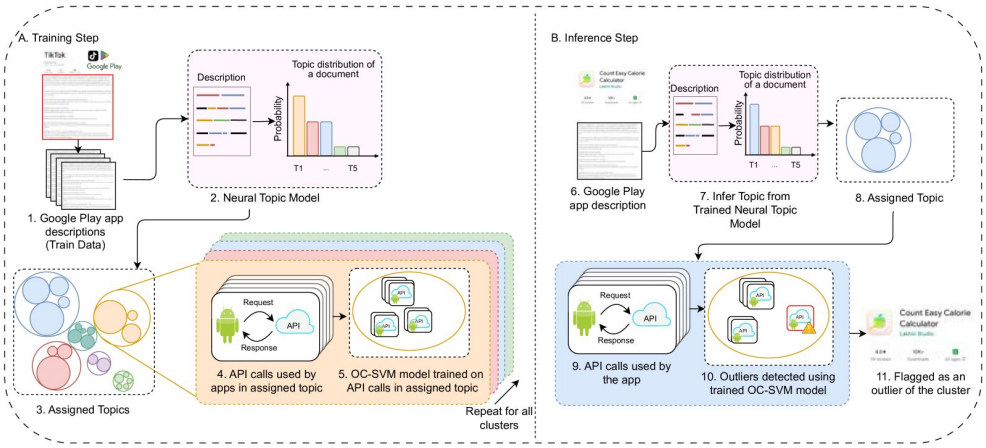


**Figure 3.1:** BERTDetect Framework

# Chapter 4

# Data Collection

## 4.1 Strategy and Challenges

This section outlines the strategy followed to collect a dataset of applications integrated with Microsoft Entra ID, which is comprehensive for the purpose of our analysis. Firstly we detail the target dataset, i.e., the specific features we are interested in collecting. Secondly we describe the possible sources from which applications can enter an Entra ID tenant, and finally we discuss the challenges faced during the data collection process.

### 4.1.1 The Target Dataset

As established in Section 1.2, the goal of this thesis consists in identifying discrepancies between M365 applications' intents and the permissions they request. To achieve this, we need to collect a dataset of applications with at least two main features:

- **Description**: a textual description of the application's purposes and functionalities, required for topic modelling.
- **List of Permissions**: the set of requested permissions, required for anomaly detection.

Other metadata such as the application name, the publisher, the M365 products the app integrates with, etc., can be useful for data exploration but they are not strictly required for our work.

### 4.1.2 Applications Sources in Entra ID

Applications can enter a tenant from multiple sources. There are applications that are visible only by admins through the Entra ID portal (*Admin Path*), and there are applications that can be added by end users (admin included) through OAuth consent (*User Path*) [55]. [1] The path of our interest is the **User Path**, since it is the

[1] we are assuming that admin consent workflow is not in place, i.e., users can directly consent to the requested permissions without admin's review

largest and the most exposed to over-privileged applications. Looking at the Figure 4.1, it is clear that users can add two types of applications to their Entra ID tenant:

- **Add-Ins**: these are the applications that extend the functionalities of built-in M365 services, such as Word, Excel, Teams, SharePoint, etc. These applications can be found in the official Microsoft AppSource store or directly from the specific marketplace within the M365 service that the add-in extends. An example of add-in is the "Wikipedia" add-in for Word and Excel which allows users to directly access wikipedia resources from the project they are working on.
- **Third-Party Applications**: these are the applications that access Entra ID resource to improve their functionalities. Example of third-party application are journaling applications like Samsung Notes, Notability, etc., that can be integrated with OneDrive to store notes over the cloud.

It's important to note that this distinction is relevant only for the design of the data collection process, since both add-ins and third-party applications are service principals within the Entra ID tenant as described in Section 2.2.

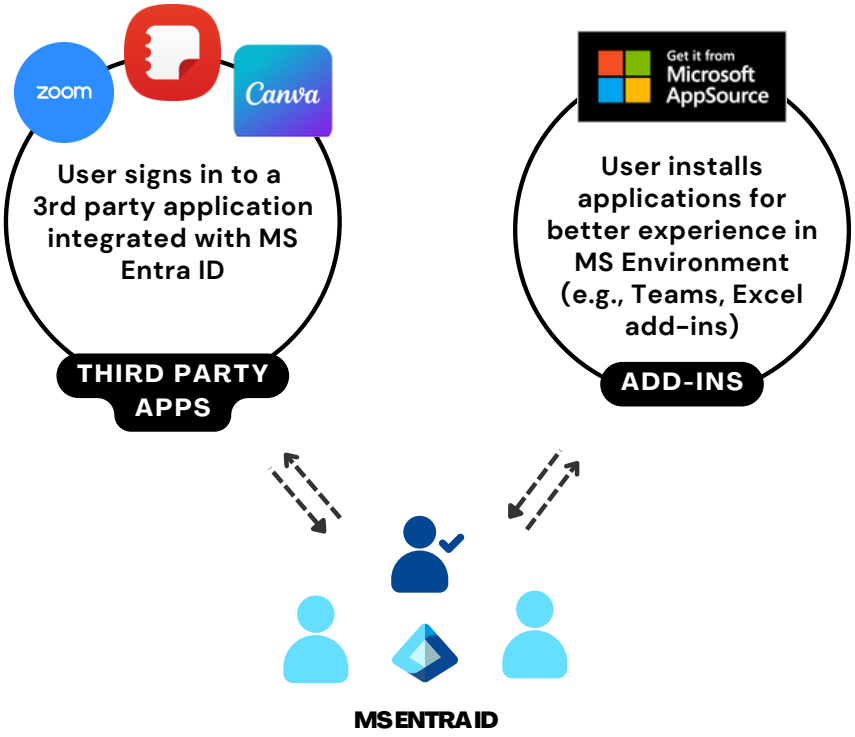


**Figure 4.1:** Applications Sources in Entra ID - User Consent Path

### 4.1.3 Data Collection Challenges

Collecting a dataset that checks the requirements described in Section 4.1.1, is not a trivial task, due to several motivations:

- the applications' manifests are not always publicly available, and even when they are, they do not always contain the list of permissions the app requests,

- the "world" of third-party applications, described in Section 4.1.2, is theoretically infinite, and lacks of a central repository where applications can be found. This makes the collection of third-party applications extremely difficult,
- permissions visibility from Entra ID portal is triggered only after the user directly consents permissions to the application, making web scraping techniques and the availability of a test Entra ID tenant necessary for the collection. An example is reported in Figure 4.2.

Additionally, the fragmented nature of the Microsoft ecosystem, composed of multiple services (Teams, Sharepoint, etc) and marketplaces (AppSource, Teams Store, Sharepoint Store, etc.), makes the job even more difficult.

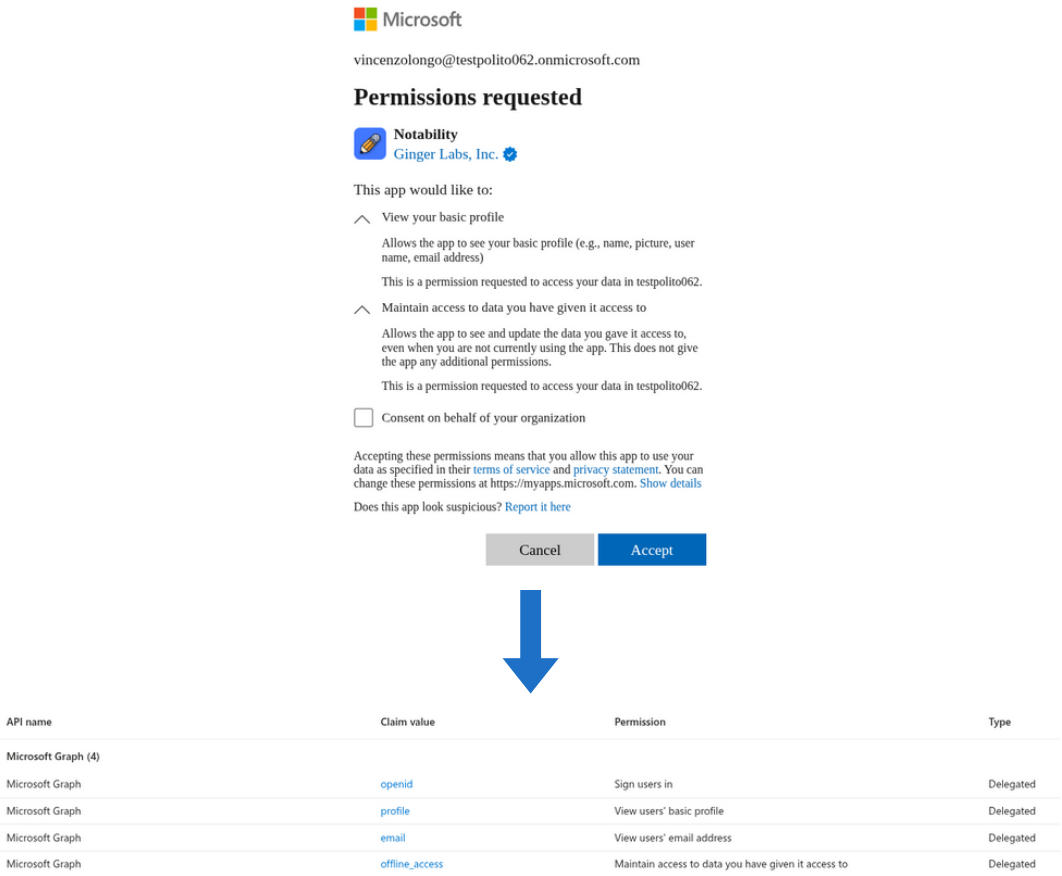


**Figure 4.2:** Example of permission prompt and permissions visibility in Entra ID Portal

## 4.2 Data Collection Process and Tools

We designed a data collection strategy that leverages multiple sources to collect a populated dataset of applications integrated with Entra ID. Firstly, we tried to find all types of public APIs that Microsoft exposes in their multiple marketplaces (AppSource, Teams Store, Sharepoint Store, etc.) to collect those applications manifests that explicitly declare their permissions. Secondly, we design a web scraping tool that

automatically accesses to a test Entra ID tenant, deploy applications from AppSource (the most complete Microsoft marketplace) and accept their permission prompts (if any) to trigger their visibility through Entra ID portal. Finally, we leveraged the PoliTO tenant application dataset to enrich the final dataset with other applications, like third-party ones, that are not present in the previous sources.

### 4.2.1 Public APIs

As the fastest and the most scalable way to collect data, we first searched for public APIs (documented or not) that Microsoft exposes in their multiple marketplaces, trying to find applications' manifests that explicitly declare the list of permissions they request.

#### Teams Store API

We found that Microsoft Graph provides an official documented API [2] to access the Teams app catalog [56]. Totally there are about 2,800 applications (the number is updated as of September 2025) listed in the catalog. Each application manifest contains both fields for the description and the list of permissions but often the permissions field is empty. It is important to remark that the permissions listed in this API are all to be intended as "Resource Specific Consent" (RSC) permissions [56, 29]. Particularly, we managed to collect 347 applications with a non-empty permissions field. The JSON object returned by the API is reported in listing A.3.

#### Sharepoint Store API

When you log in to your tenant SharePoint (`<TENANT_NAME>.sharepoint.com`), you can access the SharePoint Store [3] to add applications to your SharePoint environment. Here, when you try to add an application, the prompt shown in Figure 4.3 directly shows the list of permissions requested by the application. Specifically, given the <APP_ID> (obtained via the AppSource API 4.2.1) of a sharepoint application, it is possible to retrieve the list of permissions of the application by calling an undocumented API [4]. The API requires authentication, so it is necessary to set right cookies within the request headers. An example of the JSON object returned by this API is reported in listing A.4. It is important to notice that the type of permissions listed in this API are not reported and we marked them as "*Unknown*" in the final dataset. Unfortunately, only 221 applications out of 1222 have the permissions field not empty.

[2] `https://graph.microsoft.com/v1.0/appCatalogs/teamsApps?\protect\T1\textdollarexpand=appDefinitions`

[3] `https://<TENANT_NAME>.sharepoint.com/sites/appcatalog/_layouts/15/appStore.aspx/sharePointStore`

[4] `https://<TENANT_NAME>.sharepoint.com/sites/appcatalog/_layouts/15/storefront.aspx?task=GetAppAdditionalDetails&bm=US&cm=en-US&appid=<APP_ID>&catalog=0&uirequest=1`

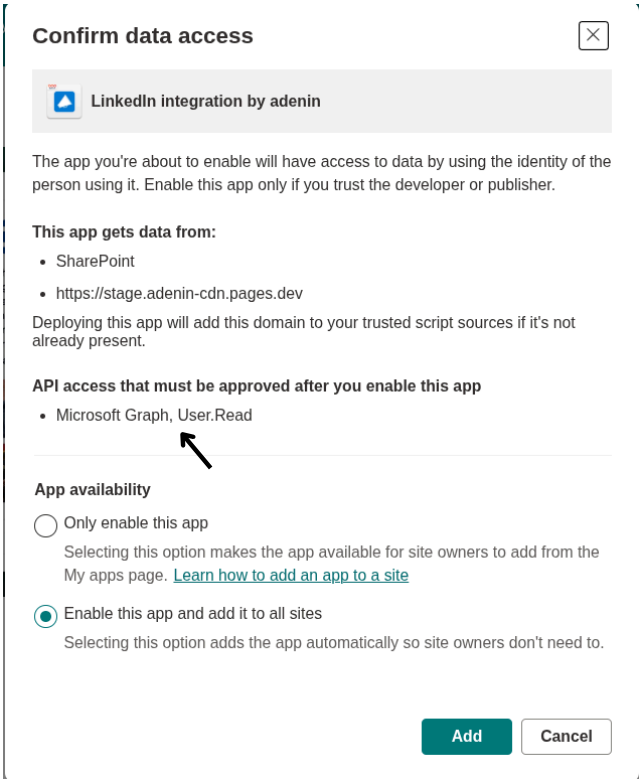


**Figure 4.3:** Example of Permission Prompt in SharePoint Store

### AppSource API

We discovered that it is possible to collect the complete manifest of the applications listed in the MS AppSource catalog by calling two undocumented public APIs. [5] The flow is the following:

1. first the `/view/tiledata` [6] MS AppSource API is called to retrieve the list of applications for a specific M365 product (e.g., Teams, Excel, SharePoint, etc.). It is possible to note that the API support pagination and filtering by product making possible the iteration over all the available applications. The number of applications per product is reported in Table 4.1. [7] Here we can retrieve the unique identifier of each application (APP_ID), which is required for the second step.

2. then, it is possible to call the `/view/app/` [8] MS AppSource API to retrieve the complete manifest of a specific application, given its unique identifier (APP_ID).

Unfortunately, the manifests retrieved through this API do not contain the list of permissions requested by the applications. On the other hand, each manifest contains

[5] The JSON objects returned by these APIs is reported in listings A.1 and A.2

[6] `https://appsource.microsoft.com/view/tiledata?ReviewsMyCommentsFilter=true&country=US&embedHost=admin-portal&entityType=App&page=<PAGE>&product=<PRODUCT>&region=ALL`

[7] The sets of applications for each product are not disjoint, i.e., some applications are listed in multiple products.

[8] `https://appsource.microsoft.com/view/app/<APP_ID>`

lots of information about the application including a well structured description, the native categories the app belongs to, the rating and popularity scores, etc. For this reason, we decided to keep this source as it would be useful for the design of the web scraping tool described in Section 4.2.2 and data enrichment.

| **Product** | **Total Apps** |
|---|---|
| Excel | 1,338 |
| Office | 412 |
| Outlook | 2,069 |
| PowerPoint | 480 |
| SharePoint | 1,222 |
| Teams | 2,595 |
| Word | 1,306 |

**Table 4.1:** Number of Applications per Product in AppSource

### 4.2.2 Web Scraping and Automated Consent

As already mentioned in Section 4.1.3, the visibility of the permissions requested by a service principal within the Entra ID portal is triggered only after the user accepts the permission prompt of the application. This adds two requirements:

- the availability of a test Entra ID tenant,
- the ability to automate the consent process for the required permissions iterating over all the applications listed in AppSource.

Opening a test Entra ID tenant resulted very easy, since Microsoft offers a free 30-days trial for Microsoft 365 E5 subscription. [9] By indicating all the required details about your organization and adding a payment method, we managed to get quite all the functionalities we need for our data collection.

For the second point, we designed a web scraping tool, called *M365 App Deployer*, that for each application listed in the AppSource catalog (Section 4.2.1) automatically accesses the AppSource page, follows all the procedures to add the application to the test tenant, accepts the permission prompt (if any) and logs a message indicating whether the operation was successful or not.

#### M365 App Deployer - Implementation Details

For the implementation of the M365 App Deployer [10] we leveraged the Selenium Browser Automation framework[11], which offers several libraries for a wide range of programming languages (Python, JavaScript, etc.) that enable and support the

[9] `https://signup.microsoft.com/get-started/signup?products=7bb264e9-90b3-41b5-a0df-202db874ea3a&mproducts=CFQ7TTC0LFLX:0022&fmproducts=CFQ7TTC0LFLX:0022&renewalterm=P1Y&renewalbillingterm=P1Y&culture=en-us&country=us&ali=1`

[10] the code is available at `https://github.com/vinclongo01/m365-app-deployer#`

[11] `https://www.selenium.dev/`

automation of web browser [57]. We implemented the tool in Python, using the `Selenium WebDriver` library which provides a set of functions that allow to drive a browser natively, as a real user would do.

The process diagram of the M365 App Deployer is shown in Figure 4.4. The requirements to run the tool are the path of the Firefox browser profile where the tenant account is already logged in and the mail of the account itself; all these should be provided by properly setting the environment variables `FIREFOX_PROFILE_PATH` and `MICROSOFT_ACCOUNT`.

Once the M365 App Deployer is started, it iterates over the list of applications got by product from the AppSource API (Section 4.2.1) and for each application it performs the following steps:

1. call the `click_get_it_now_button` function to wait for the presence of the `Get it now` button on the AppSource page of the application and click it,
2. call the `fill_account_and_sign_in` function to insert the test tenant account mail (provided within the environment variables) and sign in with it, [12]
3. call the `consent_dialog` function to wait for the presence of the consent dialog and the OAuth permissions promt (if any) and accept it,
4. call the `handle_app_deployment` function to complete the deployment process in the Entra ID Admin Center.

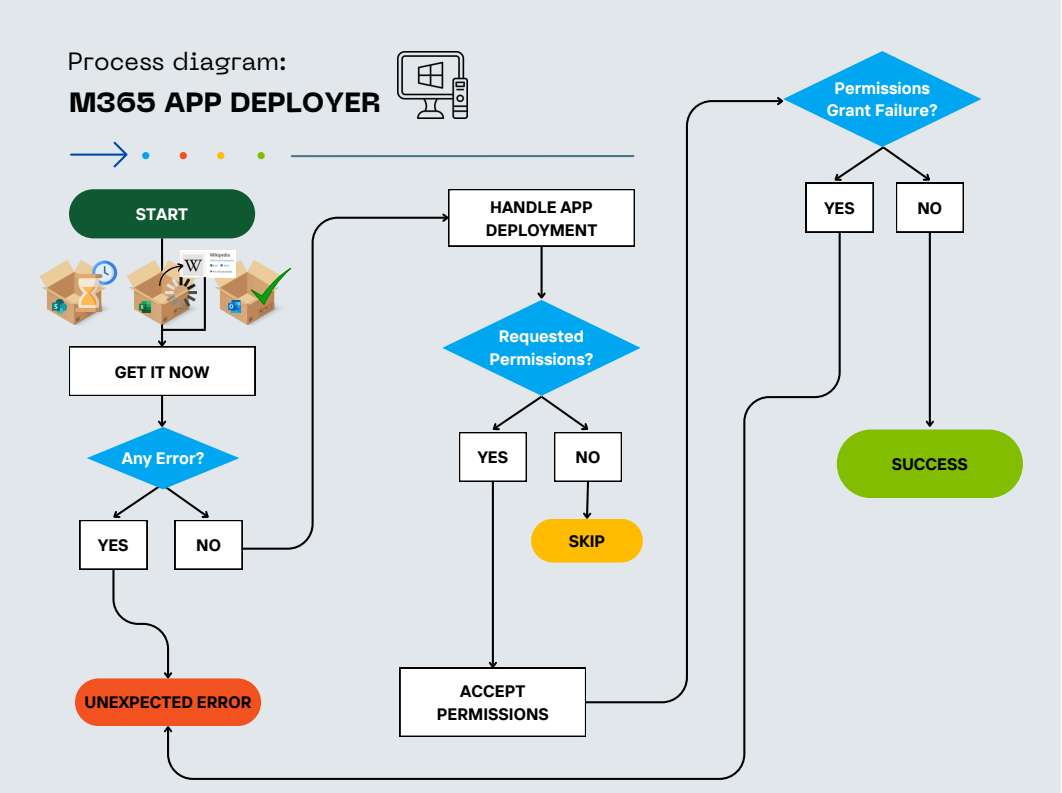


**Figure 4.4:** M365 App Deployer - Process Diagram

[12] the authentication process is skipped leveraging the fact that we load the browser profile where the test account is already logged in.

The tool is able to handle most of the exceptions that could happen during the deployment process, such as the absence of the OAuth prompt, a general timeout, etc. At the end of the deployment of each application, a log message is produced indicated the success of the operation or the type of error that happened. The most relevant output messages are reported in Table 4.2.

| **Log** | **Description** |
|---|---|
| `[SUCCESS]` | The app is deployed with accepted permissions. |
| `[ALREADY DEPLOYED]` | The app was already deployed with accepted permissions. |
| `[ERROR WHILE CLICKING ACCEPT]` | Unexpected error during OAuth Consent. |
| `[NO ACCEPT PERMISSIONS BUTTONS]` | No permissions requested while deploying the app. |
| `[GENERAL TIMEOUT]` | e.g., [TIMEOUT WAITING GET IT NOW BUTTON], not so frequent, a new attempt might help. |

**Table 4.2:** Most Relevant Output Messages of M365 App Deployer

At the end of the deployment process, we managed to deploy 330 applications out of 8,232 (about 4% of the total) listed in AppSource. This low percentage is mainly due to the fact that many deployment of applications permanently fail due to errors that the M365 App Deployer is not able to handle, such as:

- the OAuth consent unexpectedly fails due to unknown reasons (further attempts do not help),
- the application do not ask for permissions when deployed (useless for our purpose),
- the deployment process of the application goes beyond the flow the M365 App Deployer is able to handle (e.g., other pages are loaded, etc.).

### App Inventory from Entra ID Portal

Once the M365 App Deployer has finished its job, it is possible to see all the instantiated applications from the **Enterprise applications > All applications** section of the Entra ID portal. Here, for each service principal it is possible to see the API permissions they request, as already shown in Figure 4.2.

To get a complete inventory of all the service principals active in the test tenant, we leveraged the PowerShell script `app_Permissions_inventory_GraphAPI.ps1` [13] by Vasil Michev [58] which allows to create an inventory of the active applications in the tenant with a similar level of detail as surfaced within Microsoft Cloud App

[13] `https://github.com/michevnew/PowerShell/blob/master/app_Permissions_inventory_GraphAPI.ps1`

Security, without having to pay the extra license fees, including application and delegated permissions in a human-readable format. The script leverages the Microsoft Graph API and the PowerShell SDK for Microsoft Graph [59].

The final step consists in linking each row of the app inventory with the original application manifest obtained via the AppSource API (Section 4.2.1), to get the description and other metadata of the application. This is not always straightforward, as the name that we found in the app inventory is the service principal's one, which might slightly differ from the official name of the application. For instance the application "Adobe Acrobat" appears in the inventory as "Adobe Acrobat for M365". For this reason, we used a "direct then partial" match approach to associate the app inventory row with the original application manifest. Specifically, we first search for a direct match between the official name of the application (the one in the manifest) and the name of the service principal in the app inventory. If no direct match is found, we try to slightly relax the match by looking for partial matches using regular expressions allowing to ignore common suffixes and prefixes (e.g., "for M365", "-Teams", etc.).

### 4.2.3 PoliTO Tenant Dataset

The last source we used to enrich the final dataset is the application dataset of the Poltecnico di Torino tenant, containing both add-ins and third-party applications, which are not present in the previous sources. The dataset was collected in March 2024 by the PoliTO IT department and contains 958 applications. It shares the same level of detail of the Application Inventory described in Section 4.2.2. In this case we have the list of permissions for each application, but we miss the description field. As sources of descriptions we used:

- the AppSource API (Section 4.2.1 for the add-ins).
- the Microsoft Defender for Cloud Apps endpoint to get the description of third-party applications. [14]

The Official-Service Principal name discrepancies already described in Section 4.2.2 makes the association not always straightforward. Therefore, we follow the same "direct then partial" match approach used before to find the original application name and its description. At the end, we managed to keep 285 applications with both description and permissions.

## 4.3 The Resulting Dataset

After data collection from the multiple sources described in Section 4.2, we merged all the data together obtaining a dataset of 1,286 applications. The number of apps

[14] it is possible to retrieve the description plus other metadata of third-party applications integrated with Entra ID by sending an authenticated request to the endpoint `https://security.microsoft.com/apiproxy/mcas/cas/api/v1/discovery/app_catalog/` and specifying the application name within the `filter` query parameter.

per source is reported in Table 4.3.

| **Source** | **Number of Apps** |
|---|---|
| Crawler | 432 |
| PoliTO | 285 |
| SharePoint | 221 |
| Teams | 348 |

**Table 4.3:** Number of Applications Collected per Source

Before using it for our work, we cleaned the data from discrepancies and/or duplicates to obtain a final dataset of about 1000 applications with both description and list of permissions. These phases are described in the Section 4.3.1, while the final dataset characteristics are described in Section 4.3.3.

### 4.3.1 Data Merging and Preprocessing

In this phase we focus on cleaning the dataset from possible duplicates or discrepancies. Before describing the process it is important to understand that the same application can appear in multiple sources, but with different levels of detail. Unfortunately, no global unique identifier is available to easily identify duplicates, but we have to rely on two main features:

- **title**: the official name of the application, listed in any catalog (e.g., AppSource, Teams Store, etc.),
- **appName**: the name of the service principal created within the tenant, visible in the Entra ID portal.

As already mentioned in Section 4.2.3, these two names might slightly differ. Firstly, we looked for duplicates (based on the title) between the various data sources and we handle them as described in the Table 4.4.

| **Sources** | **Decision** | **Motivation** | **Quantity** |
|---|---|---|---|
| Crawler vs Teams | Merge rows | Teams' permissions are RSC (resource-specific, scoped to teams), while crawler permissions are tenant-wide. Both offer complementary perspectives. | 49 |
| Teams vs PoliTO | Drop PoliTO | Teams data is more complete and includes all declared RSC permissions. PoliTO may reflect tenant-level filtering. | 19 |
| Crawler vs PoliTO | Drop PoliTO | Crawler data reflects the full declared manifest from AppSource, while PoliTO may restrict or filter permissions based on tenant policies. | 28 |

**Table 4.4:** Data Merging Strategy for Duplicate Applications

After handling duplicates, we looked for discrepancies that could have been introduced during the "direct then partial" match approach used for description field completion (Section 4.2.2 and 4.2.3). Specifically, we looked for those applications with the same title (i.e., the same manifest) but different appName (i.e., different service principals). These cases are not always discrepancies, as the same applications might trigger the creation of multiple service principals (e.g., Zoom → "Zoom for M365" and "Zoom for Teams"). But in some cases, the association was wrong, and we had to drop some rows. After manually checking all these candidates, we dropped 61 rows with ambiguous associations.

For the topic modeling phase, we perform a basic preprocessing step consisting in filtering out applications with very short descriptions (i.e., less than 10 words) or with non-English descriptions and removing HTML tags and non-alphanumeric characters. The table below summarizes the preprocessing steps and the final number of documents used for the topic modeling.

| Preprocessing Step | Number of Apps |
|---|---|
| Initial number of apps with descriptions | 1286 |
| Handling duplicates | −96 |
| Handling discrepancies | −61 |
| Filtered out (not in English) | −40 |
| Filtered out (less than ten words) | −20 |
| Cleaned HTML descriptions | 638 |
| **Final number of apps used for topic modeling** | **1069** |

**Table 4.5:** Summary of preprocessing steps and resulting number of apps

The histogram of the description lengths (in words) is reported in Figure 4.5.

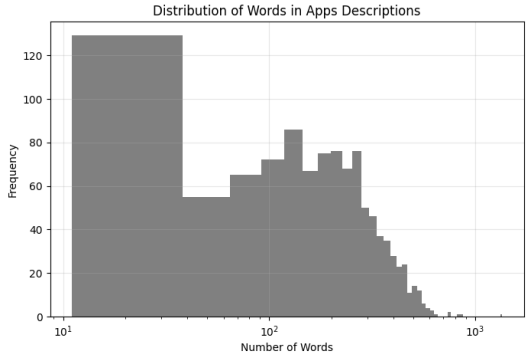


**Figure 4.5:** Histogram of Description Lengths (in words)

### 4.3.2 Risk Score Definition

As already mentioned in Section 4.1.1, a useful feature for our analysis is a risk score assigned to each application. Unfortunately we did not managed to collect a Microsoft native risk score to associate to each application of our dataset mainly due to the fragmented nature of the data sources and the lack of a common unique identifier. For this reason, we defined a custom risk score based on two indicators of exposure (IoE) [60, 61] that list the most dangerous permissions tha can compromise the security of an Entra ID tenant. We defined a function that takes the permission list of an application as input, and returns a risk score based on the following rules:

- for each requested permission add 0.5 as baseline risk,
- for each permission that is listed in the IoEs add the related score associated to the risk level of the permission.
- for each permission that is an application permissions add 5 as the scope is tenant-wide,
- for each permissions that is RSC remove 5 as the scope is limited to a specific team or chat (in case of Teams applications),

- for each read-write permission add 3 as it allows modification of data,
- for each permission that has the `.All` pattern add 5 as it allows access to all the resources of that type (e.g., `User.ReadWrite.All`).

The final risk score is the sum of all these contributions and it ranges from 0 (no risk) to higher values (higher risk). The definition of the `calculate_risk_score` function is reported in listing A.5.

### 4.3.3 Final Dataset Characteristics

At the end of the process, we obtained 1,069 applications with both description and list of permissions, plus other metadata such as risk score, title, etc. The distribution of applications per source is reported in Table 4.6.

| **Source** | **Number of Apps** |
|---|---|
| Crawler | 314 |
| PoliTO | 219 |
| SharePoint | 219 |
| Teams (merged) | 317 (49) |
| **Total** | **1,069** |

**Table 4.6:** Number of Applications in the Final Dataset per Source

# Chapter 5

# Methodology for detecting anomalous apps

The methodology consists of three phases. The first phase involves extracting topics (clusters of semantically similar applications) from the dataset using BERTopic. The second phase consists of identifying suspicious applications (i.e., requesting anomalous permissions) within each topic. The last phase is the evaluation of the results. This section provides the technical details about the topic extraction and anomaly detection techniques, as well as the evaluation methods employed to assess the models used during the experiments. The overall methodology is illustrated in Figure 5.1.

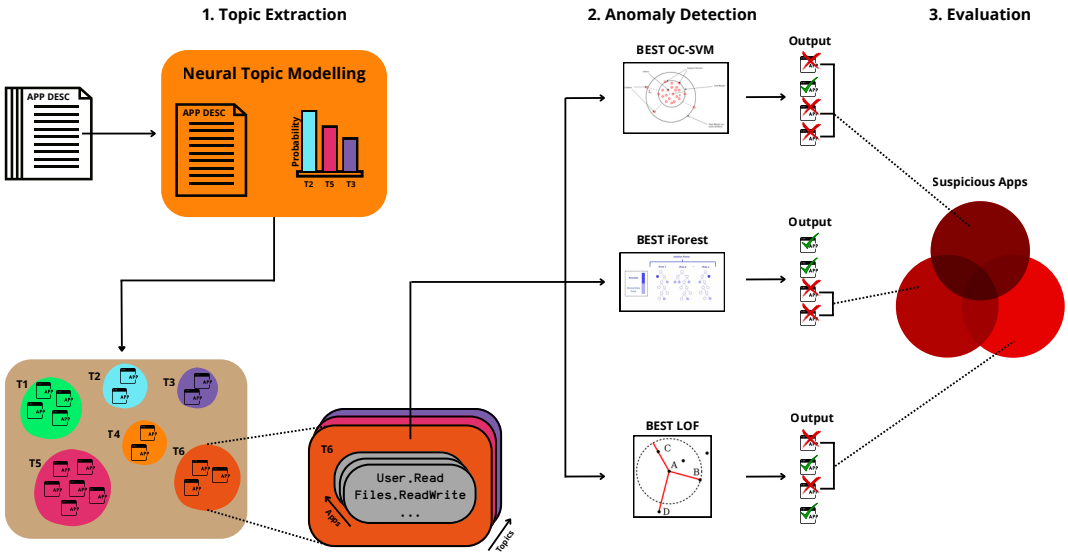


**Figure 5.1:** Overview of the methodology used in this research.

## 5.1 Phase 1 - Topic Extraction

This section describes the entire process followed to extract the most coherent topics from the dataset. It begins with an overview of BERTopic, the algorithm used for

topic extraction, and ends with the implementation details and hyper-parameter tuning process.

### 5.1.1 Topic Modelling and BERTopic

As detailed in Section 3.3, BERTopic is the current state-of-the-art topic modeling technique based on transformer models and clustering algorithms. It is highly effective in extracting coherent topics from a collection of documents being able to capture the semantic similarity among them, unlike traditional techniques. At high level, the BERTopic pipeline consists of three main phases: document embedding extraction, clustering and topic representation. Each phase consists of different steps, as shown in Figure 5.2:

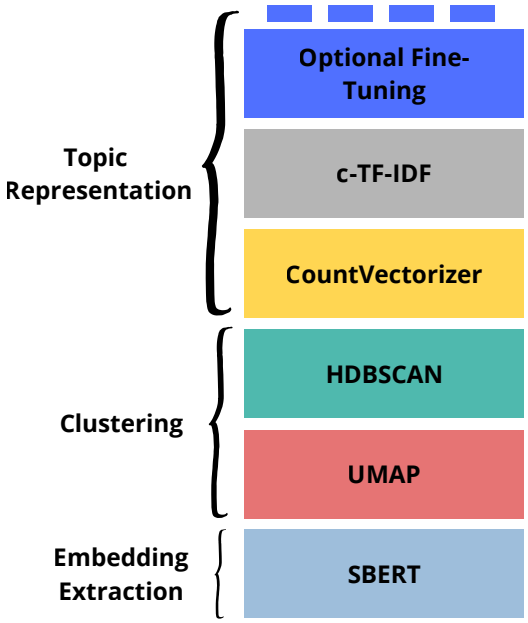


**Figure 5.2:** BERTopic Pipeline. Source: BERTopic

1. **Embedding Model**: the first step consists in the extraction of document embeddings. The default embedding model is Sentence-BERT (SBERT), which is highly capable of capturing the semantic similarity among documents.

2. **Dimensionality Reduction**: as the embedding space is usually high dimensional, it is necessary to reduce the dimensionality for the next clustering step due to the curse of dimensionality issue. The default algorithm is UMAP, which is very effective at preserving both the local and global high-dimensional space in lower dimensions.

3. **Clustering Algorithm**: the new lower-dimensional embeddings are then clustered into topics. The default algorithm is HDBSCAN, which is quite effective in capturing clusters of varying densities.

4. **Vectorizer Model**: this is the first step needed for topic representation, which is used to extract the bag of words representation of the topics.

5. **Topic Representation**: BERTopic uses a class-based Term Frequency-Inverse Document Frequency (c-TF-IDF) to extract the most representative words from the bag of words.

6. **Fine-Tune Topic Representation**: this is an optional step that allows to further refine topic representation by using GPT, T5, KeyBERT, Spacy, and other techniques.

### 5.1.2 Topic Model Implementation and Tuning

This section describes the details about the implementation and tuning of the BERTopic model. Before applying BERTopic, we performed a basic preprocessing step as described in Section 4.3.1

The objective is to extract the most coherent topics possible from the descriptions of the applications in the dataset. To achieve this, we first design a baseline configuration as a starting point, then we follow a structured "bottom-up" approach for hyperparameter tuning, and finally we validate the best configuration. The entire process is illustrated in Figure 5.3. The set of all involved hyperparameters with their descriptions is reported in Table 5.1 [62, 63, 64].

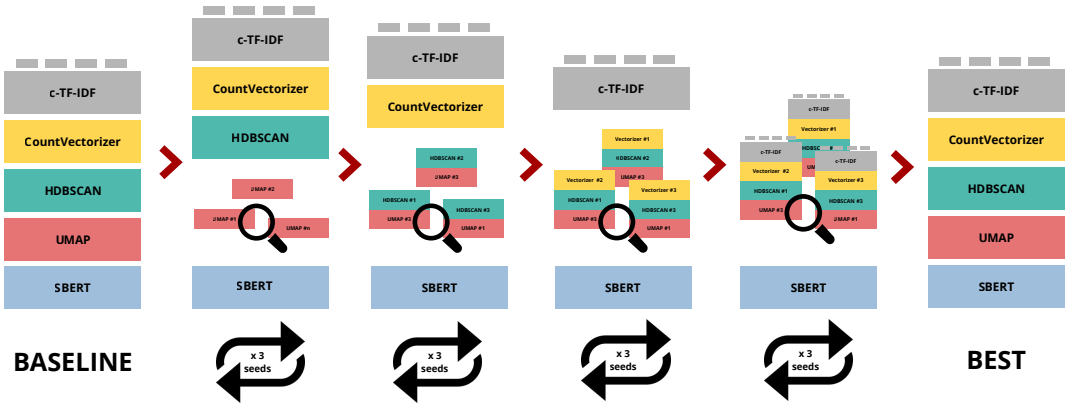


**Figure 5.3:** Bottom up approach followed for the implementation and tuning of the best BERTopic model.

| Parameter | Model | Description |
|---|---|---|
| `all-MiniLM-L6-v2` | **SBERT** | Sentence-transformer model mapping documents to a 384-dimensional vector. |
| `n_components` | **UMAP** | Number of components after dimensionality reduction. |
| `n_neighbors` | **UMAP** | Balances global vs local structure in UMAP. |
| `min_dist` | **UMAP** | Minimum distance between points in lower-dimensional space. |
| `min_cluster_size` | **HDBSCAN** | Minimum number of points forming a cluster. |
| `min_samples` | **HDBSCAN** | Minimum number of neighbors for a core point. |
| `min_df` | **CountVectorizer** | Removes too infrequent words. |
| `max_df` | **CountVectorizer** | Removes too frequent words. |
| `stop_words` | **CountVectorizer** | Set of stop-words to ignore. |
| `ngram_range` | **CountVectorizer** | Number of words packed together for c-TF-IDF. |
| `top_n_words` | **c-TF-IDF** | The number of words with highest c-TF-IDF scores to be considered as representative for each topic. Actually, this parameter is not a c-TF-IDF parameter, but we indicate as such for convenience. |

**Table 5.1:** BERTopic hyperparameters and their descriptions.

### Baseline Configuration

To establish a starting point as reference for hyper-parameter tuning we select a baseline configuration. This configuration, reported in Table 5.2, is designed choosing the most reasonable values for each parameter.

| Parameter | Base Value | Motivation |
|---|---|---|
| SBERT_model | all-MiniLM-L6-v2 | The default model for sentence embeddings in BERTopic |
| n_components | 20 | Good trade-off between dimensionality reduction and preserving structure |
| n_neighbors | 15 | Standard default balancing global vs. local structure |
| min_dist | 0.0 | Do not force separation of points, since more applications may have the same permissions features. |
| min_cluster_size | 10 | Avoids very small clusters |
| min_samples | 5 | Less than min_cluster_size to ensure core points are well-defined. |
| min_df | 2 | Filters out too infrequent words |
| max_df | 1 | Do not remove too frequent words |
| stop_words | custom_set | Remove non-informative words, using ENGLISH_STOP_WORDS and ADDITIONAL_STOP_WORDS = {"microsoft", "365", "M365", "application", "app"} |
| ngram_range | (1, 2) | Consider also bi-grams |
| top_n_words | 10 | Interpretable representation |

**Table 5.2:** Baseline BERTopic configuration with parameter values and motivations.

### Hyperparameter Tuning

The hyperparameter tuning followed the "bottom-up" approach illustrated in Figure 5.3. For each model (i.e., SBERT, UMAP, HDBSCAN, CountVectorizer, c-TF-IDF), we select the most impactful hyperparameters (if any) and we define a set of possible values that we want to explore. Then, starting from the lowest level (i.e., UMAP in our case), we iterate over all possible combinations. Each experiment is repeated using three different seeds to ensure stability of results and robustness against randomness.

For each configuration, we consider a set of metrics to evaluate clustering quality and semantic coherence [65, 50]:

- **Silhouette Score**: it indicates how appropriately a point lays within its cluster compared to other clusters. It ranges from -1 and 1 with values greater than 0.5 considered as reasonable.
- **Topic Coherence**: it measures how semantically related the representative words of a topic are. As done in BERTDetect [51], we use two different coherence metrics: $C_v$ (or CV) and Normalized Pointwise Mutual Information (NPMI) . The former ranges from 0 to 1, while the latter ranges from -1 to 1. In both

cases, higher values suggest more coherent topics. These coherence measures have been shown to better reflect the human judgement.

- **Topic Diversity**: it measures how different a topic is from the others and is defined as the percentage of unique words in the top $n$ representative words across all topics. It ranges from 0 to 1, with higher values indicating more distinct topics. The formula is:

$$\text{Topic Diversity} = \frac{|\bigcup_{k=1}^{K} T_k|}{K \cdot n} \tag{5.1}$$

  where $K$ is the number of topics, $T_k$ is the set of top $n$ words for topic $k$, and $|\cdot|$ denotes set cardinality.

- **Jaccard Redundancy**: it measures how much the topics are overlapping in terms of representative words. It is defined as follows:

$$\text{Jaccard Redundancy} = \frac{2}{K(K-1)} \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} \frac{|T_i \cap T_j|}{|T_i \cup T_j|} \tag{5.2}$$

  where $T_i$ and $T_j$ are the sets of top $n$ words for topics $i$ and $j$ respectively. It ranges from 0 to 1, with lower values indicating better separation among topics.

- **Number of Topics**: the number of clusters extracted by the model.

- **Percentage of Outliers**: the percentage of outliers found by HDBSCAN after the clustering step. It is important to note that BERTopic computes probabilities, so at inference time all the outliers are assigned to the highest probable topic. This means that no application is discarded after topic extraction. However, considering the percentage of outliers is important to evaluate the quality of the clustering, as a very high percentage may indicate a poor clustering.

After evaluating all configurations, we select the most promising ones (two-three configurations per level) and move up to the next level (i.e., HDBSCAN) and repeat the process until we reach the top level (i.e., c-TF-IDF). The exploration is made taking into account time constraints, as each configuration takes a few minutes to be evaluated and the total number of configurations is often quite high. The set of explored parameters (from bottom to top) is reported in Table 5.3. [1]

[1]The parameters that are not mentioned in the table are to be intended as fixed from the baseline configuration in Table 5.2

| Parameter | Model | Explored Values |
|---|---|---|
| n_components | **UMAP** | [20, 25, 35] |
| n_neighbors | **UMAP** | [5, 10, 15] |
| min_samples | **HDBSCAN** | [5, 10] |
| min_cluster_size | **HDBSCAN** | [5, 10, 20] |
| min_df | **CountVectorizer** | [2, 5] |
| max_df | **CountVectorizer** | [0.8, 0.95] |
| top_n_words | **c-TF-IDF** | [5, 10, 15] |

**Table 5.3:** Set of explored hyperparameters for BERTopic tuning.

#### Best Configuration Validation

After the final tuning step (i.e., c-TF-IDF), the most promising configurations are selected. To select the best one, we perform a further stability analysis by running each candidate configuration with 10 different seeds. Then we select the best configuration considering the average and standard deviation of the metrics.

Once the best configuration is selected, we perform a comparative analysis with respect to the baseline configuration to highlight the effective gain of the tuning process.

## 5.2 Phase 2 - Unsupervised Anomaly Detection within Topics

The second phase of the methodology consists in identifying anomalous applications in terms of requested permissions within each topic. We employ three distinct anomaly detection algorithms: One-Class SVM (OC-SVM), Local Outlier Factor (LOF), and Isolation Forest (iForest). Here we provide a brief overview of each algorithm and then we describe the entire pipeline followed for anomaly detection.

#### One-Class SVM

One-Class SVM is an unsupervised algorithm that sets up a decision boundary around the normal data points (assumed as the majority class), identifying any point outside this boundary as an anomaly. Before defining the boundary, OC-SVM can use a non-linear kernel function to map the data into a higher dimensional space where data is easier to separate. The most common kernel function is the Radial Basis Function (RBF). The decision boundary is defined by a hyperplane that maximizes the margin around the normal instances. OC-SVM has two main hyperparameters: $\nu$, which represents the fraction of outliers in the training data, and $\gamma$, which influences the shape of the decision boundary when using the RBF kernel. For our experiments, $\gamma$ was set to 'scale' letting the algorithm compute the best value fitting the data, as it is computed based on the number of features and the variance of the data. On the other hand, $\nu$ was tuned during the experiments [66].

### Local Outlier Factor

The Local Outlier Factor (LOF) is an unsupervised anomaly detection algorithm which operates by analyzing the local density of a point with respect to its neighbors. LOF has one major hyper-parameter, $k$, which defines the neighborood for each data point by considering the $k$-nearest distance. For each data point an anomaly score is computed, known as LOF score. This score is based on the concept of reachability and is computed as detailed in the following steps:

1. for each point $p$, its neighborood is determined by considering the $k$-nearest distance.

2. the reachability distance between two points $p$ and $o$ is computed as follows:

$$\text{reach-dist}_k(p, o) = \max\{\text{k-distance}(o), d(p, o)\} \tag{5.3}$$

   where $o$ is a neighbor of $p$.

3. the local reachability density (LRD) of a point $p$ is computed as the inverse of the average reachability distance of the point $p$ from its neighbors:

$$\text{LRD}_k(p) = \left( \frac{\sum_{o \in N_k(p)} \text{reach-dist}_k(p, o)}{|N_k(p)|} \right)^{-1} \tag{5.4}$$

   High LRD values indicates that the point is in a dense region, while low values indicates that the point is in a sparse or isolated area.

4. finally, the LOF score is computed as the average of ratios between the LRD of the the neighbors of the point $p$ and the LRD of the point $p$ itself:

$$\text{LOF}_k(p) = \frac{\sum_{o \in N_k(p)} \frac{\text{LRD}_k(o)}{\text{LRD}_k(p)}}{|N_k(p)|} \tag{5.5}$$

   The final LOF score compares the density of the considered point with the density of its neighbors. LOF values less than 1 indicates that the point is in a denser region than its neighbors, likely being an inlier, while LOF values greater than 1 indicate that the point is in a less dense region than its neighbors, likely being an anomaly.

Anomalies are then identified by setting a threshold on the LOF score, with points exceeding this threshold being classified as outliers [67].

### Isolation Forest

Isolation Forest (iForest) is an unsupervised anomaly detection algorithm which employs an ensemble of decision trees to isolate anomalies in the data. The algorithm works recursively by randomly selecting a dimension and a split value to separate data points. The partitioning of the data can be represented as a binary tree. The

process continues until every leaf of the decision tree contains a single data point. The assumption behind iForest is that anomalous samples are likely to be isolated in fewer steps with respect to normal samples, as they deviate from the majority of the data. iForest has one main hyperparameter, `n_estimators`, which defines the number of decision trees in the ensemble model. At the end, iForest computes an anomaly score for each data point defined as the average path length, $E[h(x)]$ [2], from the root (original dataset) to the leaf (isolated point) across all the decision trees in the ensemble model [3] [68].

### 5.2.1 Best Model Selection and Find Native Outliers

This section explains the entire pipeline we designed for anomaly detection within each topic. The process begins with an initial hyper-parameter tuning step to identify the best configuration for each algorithm (i.e., OC-SVM, iForest, LOF). Then we select the best model and extract the raw anomalies from each topics. At the end we evaluate the results cross checking the suspicious applications found by the three models and validating them with respect to the risk scores we computed in Chapter 4.

Instead of directly extracting anomalies from each topic, we first perform an initial hyper-parameter tuning step. As already mentioned, we perform anomaly detection within each topic, so we have one model per topic. Nevertheless, to reduce the computational cost and avoid overfitting, we perform a global search across all topics evaluating the average performance of each configuration.

The main problem in this case is that we have no prior knowledge about the anomalies in the dataset. For this reason, we design two different strategies to inject outliers in each topic: **Cross-Topic Injection** and **Synthetic Outliers Injection**.

#### Cross-Topic Injection

This strategy evaluates the models of a specific topic on real applications belonging to other topics, selected based on their permissions similarity (see Section 6.1.3). This strategy is controlled by two parameters, `min_threshold` and `max_threshold`. They define the interval of permissions similarity values to consider when selecting the topics to sample applications from. For instance, setting `min_threshold=0.2` and `max_threshold=0.4` means that we select topics whose permissions similarity with the topic under evaluation is in the range [0.2, 0.4]. Then, we randomly sample a certain number of applications from the selected topics to use as outliers.

#### Synthetic Outliers Injection

This strategy generates outliers by perturbing the permissions vectors of some real applications. Specifically, a portion of applications are randomly selected from each

[2] where $h(x)$ is the path length of a point $x$ in a single tree and $E[h(x)]$ is the expected path length across all trees in the ensemble.

[3] Finally, the score is normalized to be in the range [0, 1], with higher values indicating more anomalous points ($s(x) \sim 2^{-E[h(x)]}$).

topic. For each of them, its permission vector is then perturbated. A specified number of most common permissions (the top 30 permissions by frequency) are randomly removed, while a specified number of least common permissions (the bottom 30 permissions by frequency) are randomly added. The number of permissions to remove and add are defined by the `num_common_to_remove` and `num_rare_to_add` parameters, respectively.

### Best Model Selection

Since the algorithms selected for anomaly detection works all with thresholds, we decided to evaluate them using the Area Under the Curve (AUC) of the ROC curve, which is threshold-independent. For each algorithm (e.g., OC-SVM), injection strategy (e.g., Easy Cross-Topic), we set up a $k$-fold cross validation (with $k = 5$) to evaluate the average AUC score across all folds and topics and select the best global [4] hyper-parameter (e.g., `nu` for OC-SVM). Once the best hyper-parameter is selected (e.g., `nu=0.1` for OC-SVM), we evaluate the performance of the models by plotting the average ROC curve across all topics. It is important to note, that each strategy and level of difficulty may have different best hyper-parameters for each algorithm. At this point we select the best injection strategy for each algorithm (e.g., Medium Synthetic Outliers for OC-SVM), considering both the AUC score and the realism of the strategy, fixing the best hyper-parameters found during the grid search. The process is illustrated in Figure 5.4.

[4] the grid search is performed globally, i.e., find the hyperparameter which is optimal for all topics

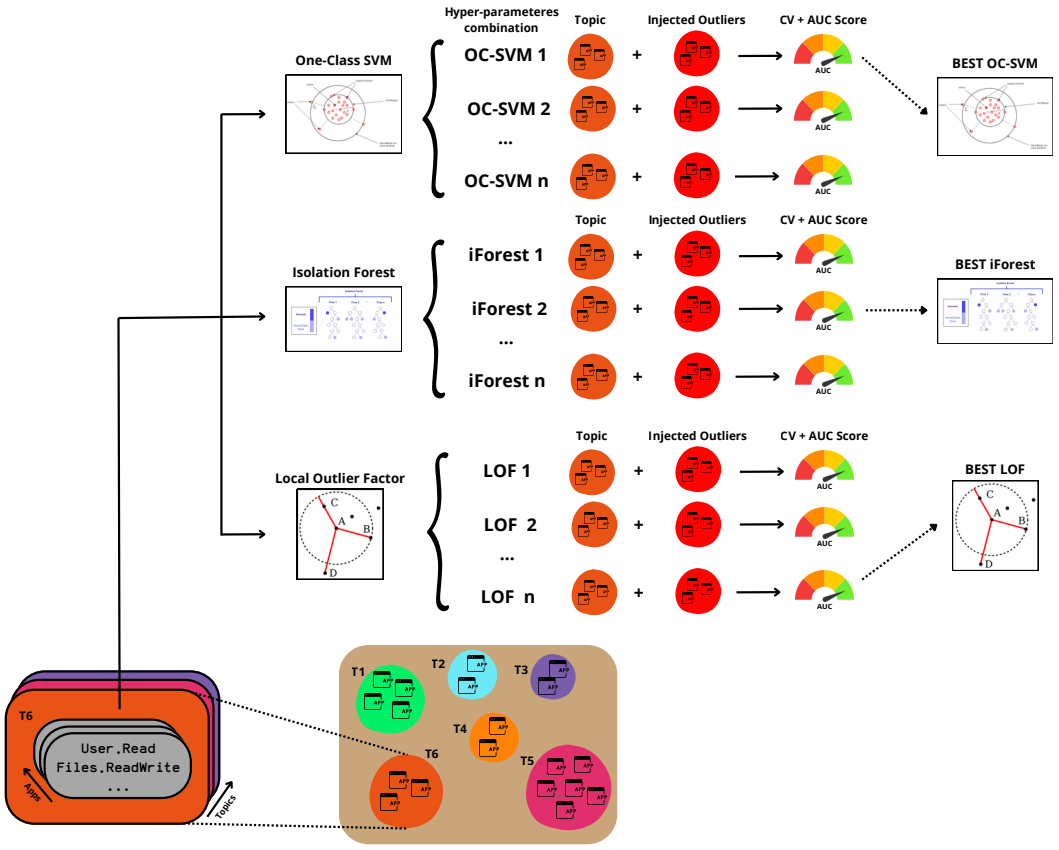


**Figure 5.4:** Best Models Selection Pipeline.

Subsequently, we set the stage for the native outlier identification task by training the best model on the whole topic and evaluating it on the injected outliers (i.e., no cross validation). In this way we can evaluate both the ability of the model in identifying the injected outliers and the portion of native outliers identified in each topic. At the end, each identified outlier is labelled as `is_<MODEL>_outlier` in the dataset for the next evaluation phase.

## 5.3 Phase 3: Evaluation Methodology

This section outlines the methodology to validate the results obtained from the previous phases. Specifically, we divided the evaluation into three main steps:

- **Topic-Less Approach**: we define a baseline OC-SVM model trained on the whole dataset without considering the concept of topics. This allows us to evaluate the effectiveness of the topic-based approach.
- **Cross-Model Analysis**. we analyse the portion of applications identified as outliers by all the models (both topic-based and topic-less). Then we evaluate the level of agreement plotting a pairwise comparision matrix among the models and computing a self-defined agreement score. The score is computed at application level and then averaged across all applications. It ranges from 0 to 1 with higher values indicating more agreement among the models. It

basically is a min-max normalization of the number of models that agree on the application being an outlier. The score is computed as follows:

$$\text{Agreement Score} = \begin{cases} 1 & \text{if } c = 0 \\ \frac{c-1}{k-1} & \text{otherwise} \end{cases} \tag{5.6}$$

where $c$ is the number of models that agree on the application being an outlier, and $k$ is the total number of models (in our case $k = 4$). The score is 0 when only one model classifies the application as outlier, and 1 when all models agree or none of them do.

- **Validation Against Self-Defined Risk Score**: we validate the suspicious applications found by the models with respect to the risk scores computed in Chapter 4. This allows us to assess the effectiveness of the models in identifying high-risk applications.

# Chapter 6

# Results

This chapter presents the results obtained from the implementation of the methodologies described in Chapter 5. It begins with the topic extraction process, including the selection of the best configuration for the BERTopic model and the analysis of the extracted topics. The chapter then moves to the anomaly detection phase, presenting the models performance on the injected outliers, and providing insights into the native outliers identified within each topic. Then, it presents an evaluation of the results obtained from cross-checking the findings from each model, validating them against the self-defined risk score, and comparing them with a topic-less baseline approach. The chapter concludes with case studies of applications identified as suspicious by our approach, providing a qualitative interpretation of the outcome, highlighting both the strengths and the limitations of our work.

## 6.1 Topic Modeling Results

This section presents the results of the topic modelling process, starting with the tuning and selection of the best BERTopic configuration, followed by the analysis of the extracted topics.

### 6.1.1 Baseline Model Evaluation

As detailed in Section 5.1 we start by evaluating the performance of a baseline BERTopic model (described in Table 5.2). We run the model 10 times with different random states to assess the stability of the results against randomness. The results are summarized in the following table:

| Metric | Mean | Std | Interpretation |
|---|---|---|---|
| Silhouette Score | 0.55 | 0.05 | Good cluster cohesion |
| Coherence (CV) | 0.41 | 0.04 | Unsatisfactory semantic consistency |
| Coherence (NPMI) | −0.20 | 0.01 | Weak semantic alignment |
| Diversity Score | 0.76 | 0.03 | Good uniqueness across topics |
| Jaccard Redundancy | 0.02 | 0.02 | Low topic overlap |
| Nr. Topics (effective) | 29 | 2 | |
| % Outliers | 21.42 | 2.40 | Moderate outlier rate |

**Table 6.1:** Performance metrics of baseline BERTopic model

Based on these results, we see that while the clustering quality is good, the semantic coherence can be improved. The Complementary Cumulative Distribution Function (CCDF) of topics coherence scores for the baseline model is shown in Figure 6.1. This plot illustrates the portion of topics with coherence scores above the threshold on the x-axis [1].

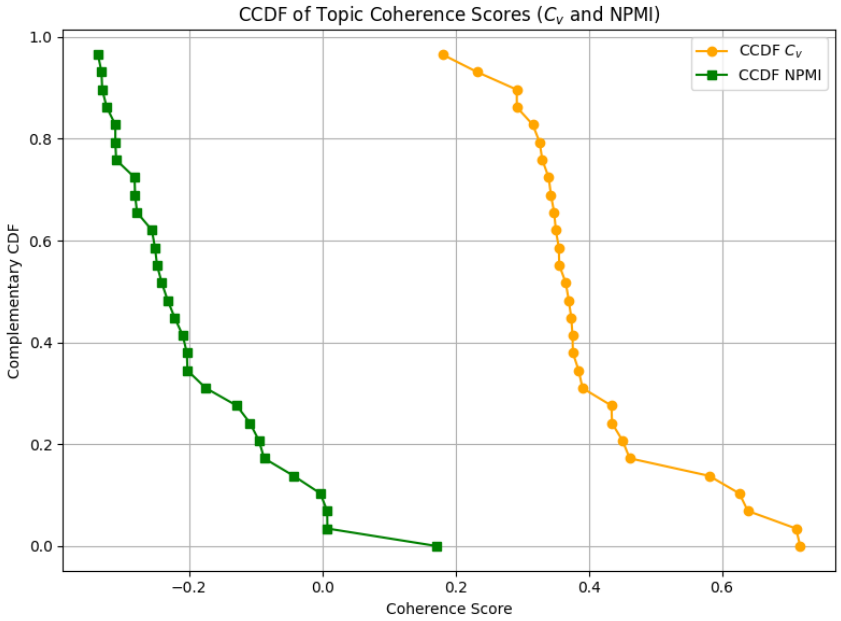


**Figure 6.1:** CCDF of topics coherence scores for baseline BERTopic model

It confirms the observations from the table, with about 70% of topics having a CV coherence score lower than 0.4 and a NPMI coherence score lower than -0.2. Therefore, the main goal to achieve in the next hyper-parameter tuning step is to improve the coherence scores across topics while maintaining a good clustering quality.

### 6.1.2 Bottom-Up Hyper-parameter Tuning

As described in Section 5.1.2 and depicted in Figure 5.3, we perform a "bottom-up" hyper-parameter tuning process selecting the set of parameters which can have the

[1] after running the baseline model with `random_state=42`

highest impact on the model performance. We proceeded with the tuning process starting from one model and running each configuration of parameters with three different random states. At each step we select the best configurations to include in the next search for the upper model and its parameters, and so on. Now we present the results at each step of the process, with the configurations tested and the best ones selected for the next step until we reach the final model.

### UMAP Hyper-parameter Tuning

We started the tuning process with UMAP, testing the following configurations keeping the other models with their values as in the baseline model (see Table 5.2).

| **Parameter** | **Explored Values** |
|---|---|
| n_components | [20, 25, 35] |
| n_neighbors | [5, 10, 15] |

**Table 6.2:** UMAP hyper-parameters and explored values

We expected to have more impact on the clustering quality metrics (i.e., Silhouette score, % Outliers, Nr. Topics) since UMAP is responsible of extracting the latent structure of data that will be then clustered by HDBSCAN. We mainly focus our attention on % Outliers and Silhouette score, since the number of topics can be adjusted later by tuning HDBSCAN parameters. The results of this tuning step are summarized in Figure 6.2 by plotting for each metric and random state, the values obtained for each configuration in a matrix format, where each row corresponds to a value of `n_components` and each column to a value of `n_neighbors`. The matrices on the same column correspond to the same random state, while the matrices on the same row provide insights about the same metric across different random states.

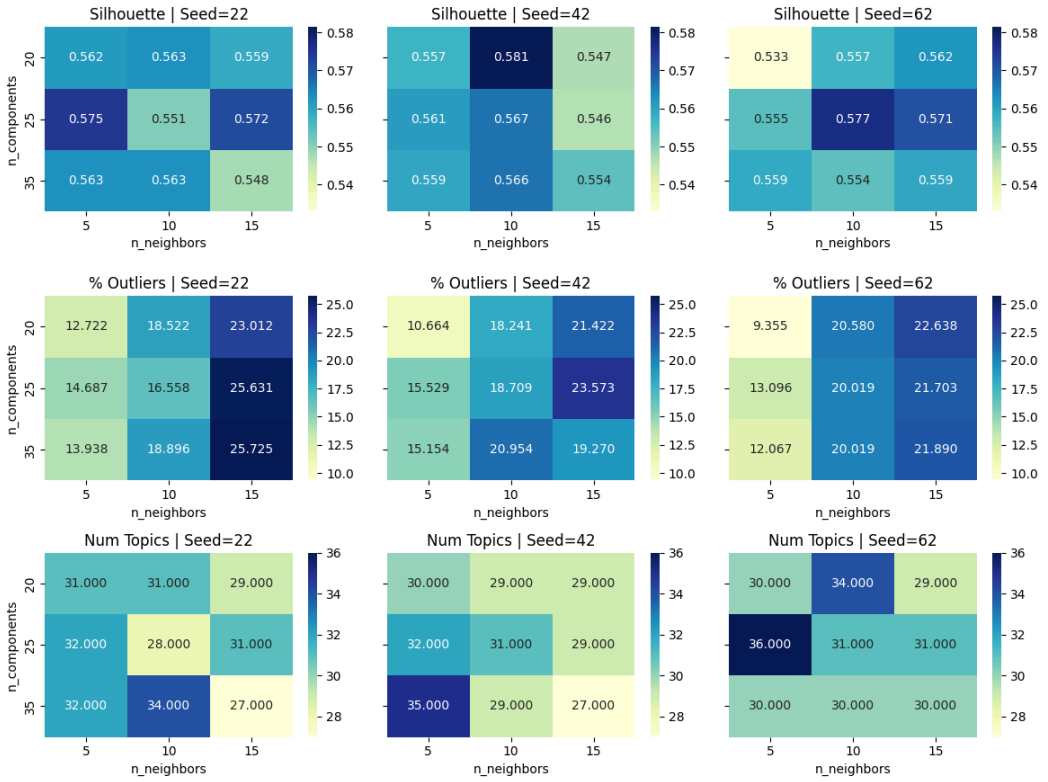


**Figure 6.2:** Results of UMAP hyper-parameter tuning: Silhouette Score, % Outliers, Nr. Topics

Looking at the results, we can see that increasing the number of neighbors tends to increase the number of outliers, while the number of components does not have a clear trend. Based on these observation, we decided to discard configurations with `n_neighbors=15`. Then, focusing on the Silhouette score, we selected the most promising configurations in terms of performance and stability across the three random states. The selected configurations are reported in Table 6.3. Regarding the topic coherence metrics, Figure 6.3 shows that as expected they are not significantly affected by UMAP parameters, as they show similar distributions across configurations.

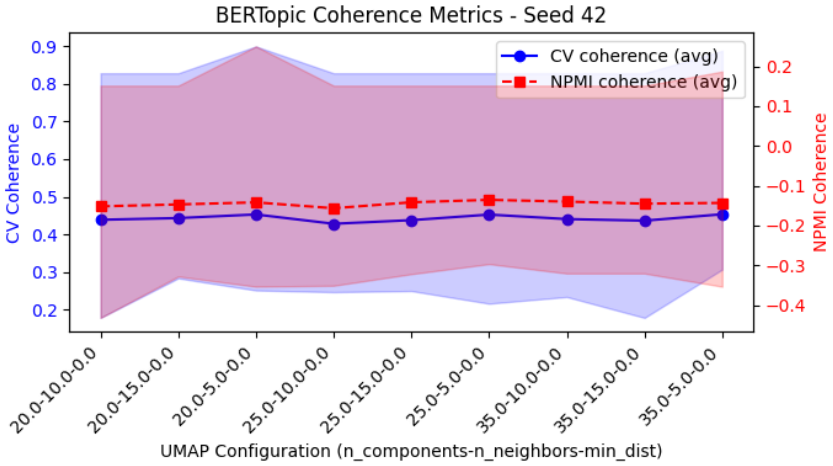


**Figure 6.3:** Results of UMAP hyper-parameter tuning: Coherence (CV and NPMI) with `random_state` = 42

| **Configuration** | `num_components` | `n_neighbors` |
|---|---|---|
| A | 20 | 10 |
| B | 25 | 5 |
| C | 35 | 5 |

**Table 6.3:** Selected UMAP configurations for further tuning

### HDBSCAN Hyper-parameter Tuning

We then proceeded with searching for the best HDBSCAN configuration, while varying the previously selected UMAP configurations and keeping the other models with their baseline values [2]. The space of explored parameters is reported in Table 6.4.

| **Parameter** | **Explored Values** |
|---|---|
| min_cluster_size | [5, 10, 20] |
| min_samples | [5, 10] |
| UMAP_config | [A, B, C] |

**Table 6.4:** HDBSCAN hyper-parameters and explored values

Again, our focus is on clustering quality and stability across random states. We follow the same approach as before, plotting results in a matrix format for each metric and random state. The results are shown in Figure 6.4.

[2] the general process is depicted in Figure 5.3

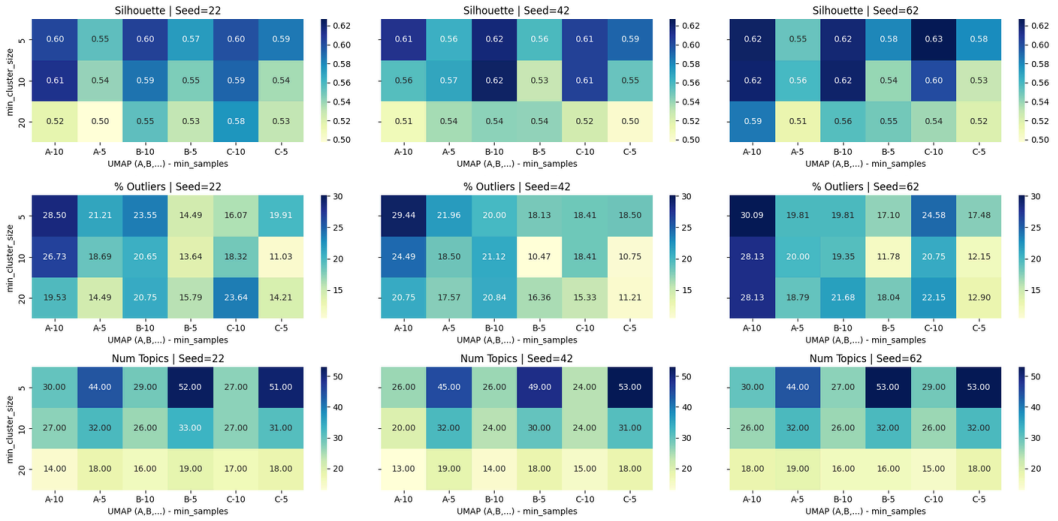


**Figure 6.4:** Results of HDBSCAN hyper-parameter tuning: Silhouette Score, % Outliers, Nr. Topics

We can see that `min_cluster_size = 5` tends to produce a very high number of topics (up to 53) which we considered unreasonably high given the size and the nature of the dataset. The combination A-10 (i.e., UMAP configuration A with `min_samples=10`) tends to produce a high number of outliers with less stable results across random states. For this reason, we discard all the configurations with `min_cluster_size=5` or the UMAP-`min_samples` combination A-10. Finally, focusing on the Silhouette score, we select the most promising configurations as before, reported in Table 6.5. Figure 6.5 confirms that the topic coherence metrics remain stable across configurations and random states, as expected.

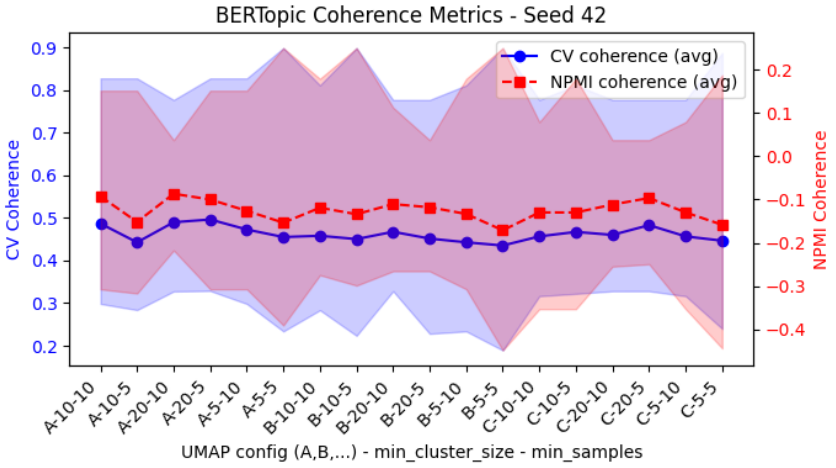


**Figure 6.5:** Results of HDBSCAN hyper-parameter tuning: Coherence (CV and NPMI) with `random_state` = 42

| Parameter - Model | Config A | Config B |
|---|---|---|
| n_neighbors - UMAP | 5 | 5 |
| n_components - UMAP | 25 | 35 |
| min_cluster_size - HDBSCAN | 10 | 10 |
| min_samples - HDBSCAN | 10 | 10 |

**Table 6.5:** Selected UMAP-HDBSCAN configurations and their model parameters

### Vectorizer Hyper-parameter Tuning

The next step consists of tuning the vectorizer's parameters, while varying the two selected UMAP-HDBSCAN configurations. For the vectorizer model we used a lemmatized version of the `CountVectorizer`. This approach allows to have a better representation of the topics by reducing co-occurrences of different forms of the same word (e.g., "calendar" and "calendars"). The explored parameters are reported in Table 6.6.

| Parameter | Explored Values |
|---|---|
| min_df | [2, 5] |
| max_df | [0.8, 0.95] |
| UMAP-HDBSCAN config | [A, B] |

**Table 6.6:** CountVectorizer and UMAP-HDBSCAN hyper-parameters and explored values

In this case, we expect to have more impact on topics coherence metrics, as the vectorizer influences the representation of topics through the c-TF-IDF scores. The results are shown in Figures 6.6 and 6.7.

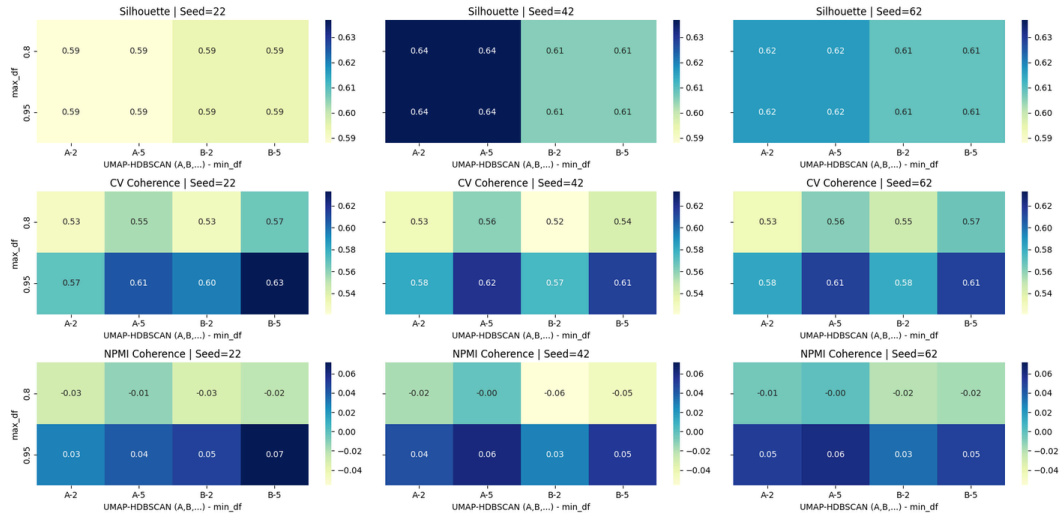


**Figure 6.6:** Results of CountVectorizer hyper-parameter tuning: Coherence (CV and NPMI) and Silhouette Score

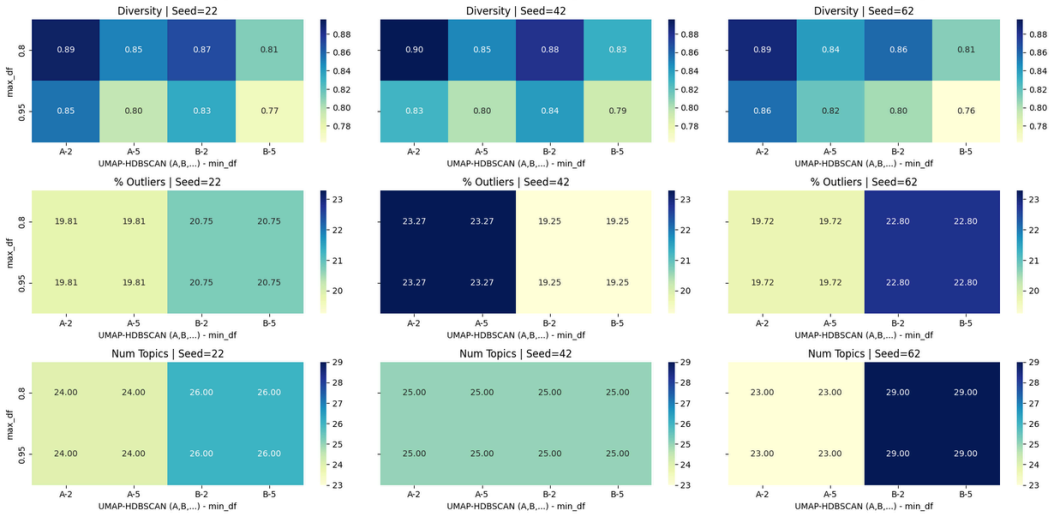


**Figure 6.7:** Results of CountVectorizer hyper-parameter tuning: % Outliers, Nr. Topics, Diversity Score

As expected, we can see that for fixed UMAP-HDBSCAN configuration, the silhouette score, % Outliers, and Nr. Topics are constant with respect to vectorizer parameters, with only slight variations across random states. On the other hand, considering topic coherence metrics, we observed that discarding words that appear in more than 80% of documents (i.e., `max_df=0.8`) is too restrictive for our dataset. For this reason, we decided to discard all configurations with `max_df=0.8` and keep the other four configurations for the next final search. The diversity score is more influenced by vectorizer parameters, but we focus on it in the next step, when we will decide the best `top_n_words` parameter for c-TF-IDF, which directly impacts on topic diversity. The selected configurations are reported in Table 6.7.

| **Parameter - Model** | **A** | **B** | **C** | **D** |
|---|---|---|---|---|
| n_neighbors - UMAP | 5 | 5 | 5 | 5 |
| n_components - UMAP | 25 | 35 | 25 | 35 |
| min_cluster_size - HDBSCAN | 10 | 10 | 10 | 10 |
| min_samples - HDBSCAN | 10 | 10 | 10 | 10 |
| min_df - Vectorizer | 2 | 2 | 5 | 5 |
| max_df - Vectorizer | 0.95 | 0.95 | 0.95 | 0.95 |

**Table 6.7:** Selected UMAP-HDBSCAN-Vectorizer configurations for final model search

### c-TF-IDF Hyper-parameter Tuning

Finally, we tuned the `top_n_words` parameter of BERTopic, which directly impacts on coherence and diversity scores. As specified in Section 5.1.2, we denoted the `top_n_words` parameter as a c-TF-IDF parameter for convenience, as the top-n representative words for each topic are selected based on their c-TF-IDF scores.

Again, we vary the previously selected UMAP-HDBSCAN-Vectorizer configurations along with `top_n_words` values. The explored parameters are reported in Table 6.8.

| **Parameter** | **Explored Values** |
|---|---|
| top_n_words | [5, 10, 15] |
| UMAP-HDBSCAN-Vectorizer config | [A, B, C, D] |

**Table 6.8:** c-TF-IDF and UMAP-HDBSCAN-Vectorizer hyper-parameters and explored values

The results are shown in Figure 6.8.

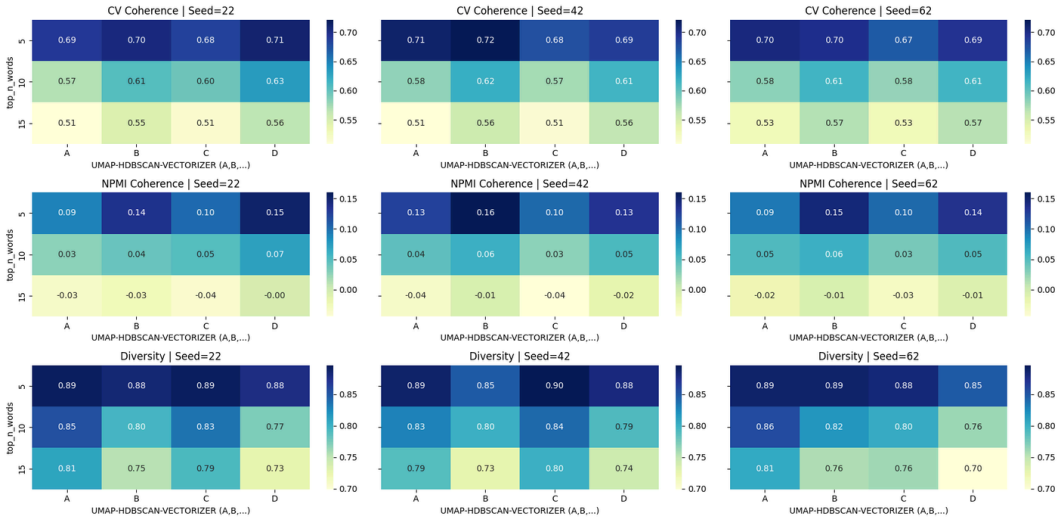


**Figure 6.8:** Results of c-TF-IDF hyper-parameter tuning: Coherence (CV and NPMI) and Diversity Score

In our experiments we observed that increasing the number of representative words per topic tends to introduce more noise, impacting negatively on both coherence and diversity scores. The best performance was obtained with `top_n_words=5` and we selected all configurations with this value for the final validation step.

### Final Model Validation and Selection

At this point, we obtained a total of 4 configurations as the most promising ones, reported in Table 6.9.

| Parameter - Model | A | B | C | D |
|---|---|---|---|---|
| n_neighbors - UMAP | 5 | 5 | 5 | 5 |
| n_components - UMAP | 25 | 35 | 25 | 35 |
| min_cluster_size - HDBSCAN | 10 | 10 | 10 | 10 |
| min_samples - HDBSCAN | 10 | 10 | 10 | 10 |
| min_df - Vectorizer | 2 | 2 | 5 | 5 |
| max_df - Vectorizer | 0.95 | 0.95 | 0.95 | 0.95 |
| top_n_words - c-TF-IDF | 5 | 5 | 5 | 5 |

**Table 6.9:** Final selected BERTopic configurations

The final validation step consists in evaluating the stability of each configuration across 10 different random states. The results are summarized in Tables 6.10 and 6.11.

| Config | Silhouette Score | Nr. Topics | Outliers (%) |
|---|---|---|---|
| A | $0.590 \pm 0.022$ | $25.3 \pm 2.0$ | $19.6 \pm 2.4$ |
| B | $\mathbf{0.597 \pm 0.015}$ | $\mathbf{25.6 \pm 1.2}$ | $\mathbf{19.8 \pm 2.7}$ |
| C | $0.590 \pm 0.022$ | $25.3 \pm 2.0$ | $19.6 \pm 2.4$ |
| D | $\mathbf{0.597 \pm 0.015}$ | $\mathbf{25.6 \pm 1.2}$ | $\mathbf{19.8 \pm 2.7}$ |

**Table 6.10:** Clustering performance metrics for the final BERTopic configurations (mean ± std over 10 seeds). It is evident that configurations with same UMAP-HDBSCAN parameters yield identical clustering metrics. Bold values indicate the best performance.

| Config | CV Coherence | NPMI Coherence | Diversity | Jaccard Redundancy |
|---|---|---|---|---|
| A | $0.652 \pm 0.020$ | $0.092 \pm 0.023$ | $0.882 \pm 0.015$ | $0.008 \pm 0.001$ |
| B | $0.657 \pm 0.022$ | $0.104 \pm 0.032$ | $\mathbf{0.878 \pm 0.015}$ | $\mathbf{0.008 \pm 0.001}$ |
| C | $0.681 \pm 0.024$ | $0.131 \pm 0.031$ | $0.849 \pm 0.018$ | $0.011 \pm 0.003$ |
| D | $\mathbf{0.689 \pm 0.026}$ | $\mathbf{0.139 \pm 0.030}$ | $0.852 \pm 0.018$ | $0.010 \pm 0.003$ |

**Table 6.11:** Summary of topic coherence and diversity metrics for the final BERTopic configurations (mean ± std over 10 seeds). Bold values indicate the best performance.

The configuration D shows the best trade-off between clustering quality and topic coherence. For this reason it was selected as the final BERTopic configuration for our analysis.

### 6.1.3 Analysis of Extracted Topics

This section presents the overview and analysis of the extracted topics after applying the best BERTopic configuration selected in the previous section (i.e., configuration

D in Table 6.9 with `random_state=42`). The high level statistics of the topic model compared to the baseline model are summarized in Table 6.12.

| **Metric** | **Baseline** | **Selected Configuration** |
|---|---|---|
| Silhouette | 0.55 | 0.59 ↑ |
| Nr. Topics | 29 | 24 ↓ |
| % Outliers | 21.42% | 19.83% ↓ |
| CV Coherence (mean) | 0.41 | 0.65 ↑ |
| NPMI Coherence (mean) | −0.20 | 0.10 ↑ |
| Diversity | 0.76 | 0.86 ↑ |
| Jaccard (Redundancy) | 0.02 | 0.01 ↓ |

**Table 6.12:** Comparison of topic modeling metrics between baseline and selected BERTopic configuration.

At a high level, we can see that the selected configuration outperforms the baseline model in all the considered metrics. The improvements obtained in terms of topic coherence are particularly significant with about 80% of topics having a CV coherence score higher than 0.6 and a positive NPMI coherence score. The CCDF and the distribution of topics coherence scores for the selected configuration compared to the baseline model are shown in Figure 6.9.

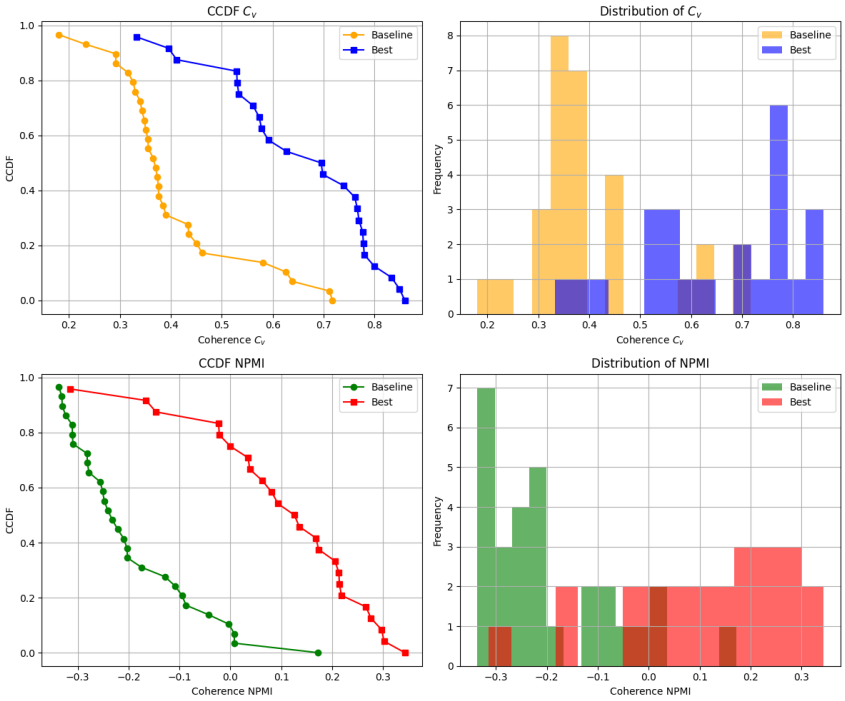


**Figure 6.9:** CCDF and Distribution of topics coherence scores: Baseline vs Best Configuration

### Topics Overview

The final model identified a total of 24 topics. The distribution of applications across topics is a somewhat unbalanced, with about 40% of topics with more than 50 applications, while 30% of topics contain fewer than 30. However, having topics with a small number of applications is expected given the characteristics of our dataset, with a limited number of applications and a reasonable variety of functionalities. The distribution of applications across topics is shown in the histogram and the Empirical Cumulative Distribution Function (ECDF) in Figure 6.10.

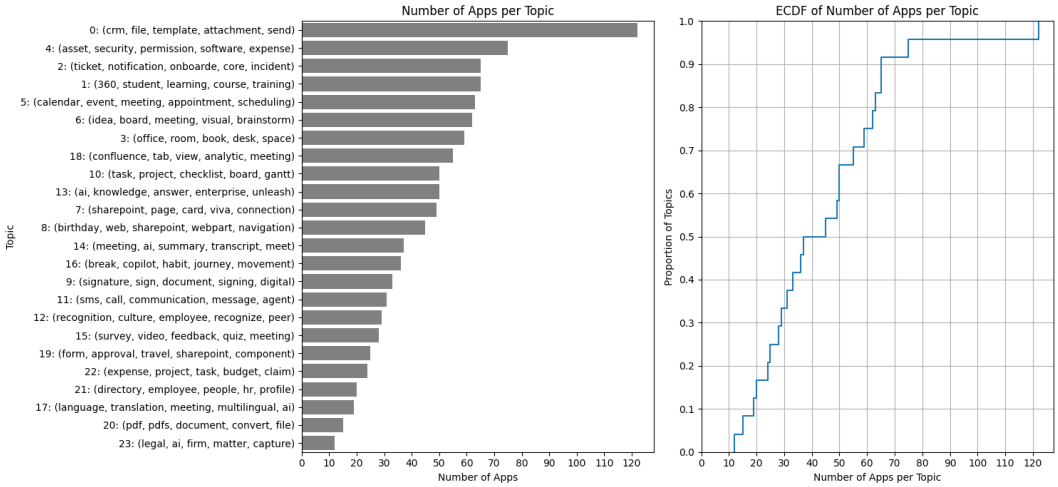


**Figure 6.10:** Distribution of applications across topics: Histogram (left) and ECDF (right)

### Topics Interpretation

As already mentioned, BERTopic assigns a label to each topic based on the top-5 representative words extracted using the c-TF-IDF scores. The complete list of topics along with their top-5 representative words and number of applications is reported in the Appendix B.1. The extracted topics reflect the nature of applications designed for Microsoft 365 services, as the most populated topics are related to productivity and collaboration. In this section we provide a qualitative interpretation of the top four and the bottom four topics in terms of number of applications populating them.

The word clouds of the top four topics in terms of number of applications suggest the following themes:

- **File Management, 122 apps**: applications related to file storage, sharing, and handling.
- **Security, 75 apps**: applications focused on asset management and protection.
- **Notifications, 65 apps**: applications that provides notifications and alerts to users.
- **E-learning, 65 apps**: applications for online learning and educational purposes.

The word clouds of these topics are shown in Figure 6.11.


**Figure 6.11:** Word clouds of the top four topics in terms of number of applications

On the other hand, the less populated topics are more specific and represent niche areas of the world of the Microsoft 365 applications in our dataset, such as:

- **Human Resources, 20 apps**: applications focused on human resources management, recruitment and life cycle handling.
- **AI Translation, 19 apps**: applications focused on translation services using artificial intelligence.
- **PDF Management, 15 apps**: applications focused on PDF document management and manipulation.
- **AI Legal Assistant, 12 apps**: applications designed for legal assistance and document review using artificial intelligence.

The word clouds of these topics are shown in Figure 6.12.


**Figure 6.12:** Word clouds of the four less populated topics in terms of number of applications

The Table in the Appendix B.1 additionally provides the interpretation of each topic based on the analysis of the representative words and the applications populating them.
Considering the permissions required by the applications in these topics, we can observe consistency with their intents. For instance, as shown in Figure 6.13, applications in Topic 0 ("crm_file_template_attachment_send") related to file management tend to require permissions specific for file sharing and management (e.g, `Mail.ReadWrite.All`, `Files.ReadWrite.All`) which are consistent with the theme of the topic, while least frequent permissions are mainly related to more specific functionalities such as teams and chat management (`Chat.ReadWrite.All`, `TeamsSettings.Read.All`). Obviously, being in the least required permissions does

not mean that these permissions are not needed for the application that require them. However, as the case of the Topic 0, finding in this topic an application requiring the High Privileged `Directory.ReadWrite.All` permission is quite rare and suspicious.

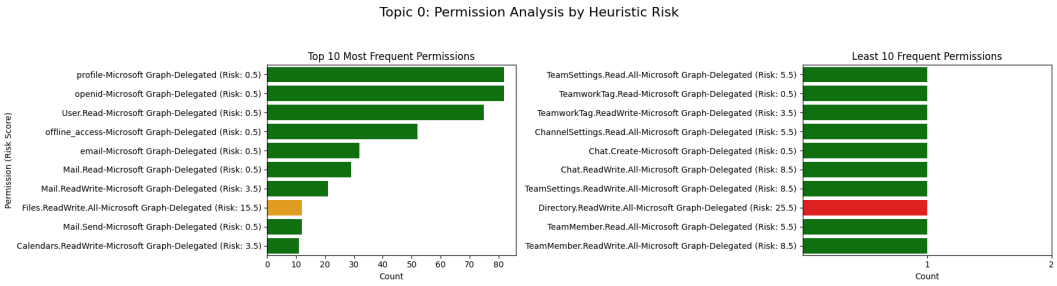


**Figure 6.13:** Top 10 (left) and Least 10 (right) permissions required by applications in Topic 0 ("crm_file_template_attachment_send")

### Topics Similarities

Now we focus on the relationships between topics, by analyzing their similarities based on semantic representations and permissions required by the applications within each topic.

The semantic similarities are computed considering both the Topic Embeddings and the c-TF-IDF Vectors, and extracting the pairwise cosine similarities between topics. We visualize the results in a heatmap format, ordering topics based on the Hierarchical Clustering provided by BERTopic (see Appendix B.1). The heatmaps are shown in Figure 6.14.

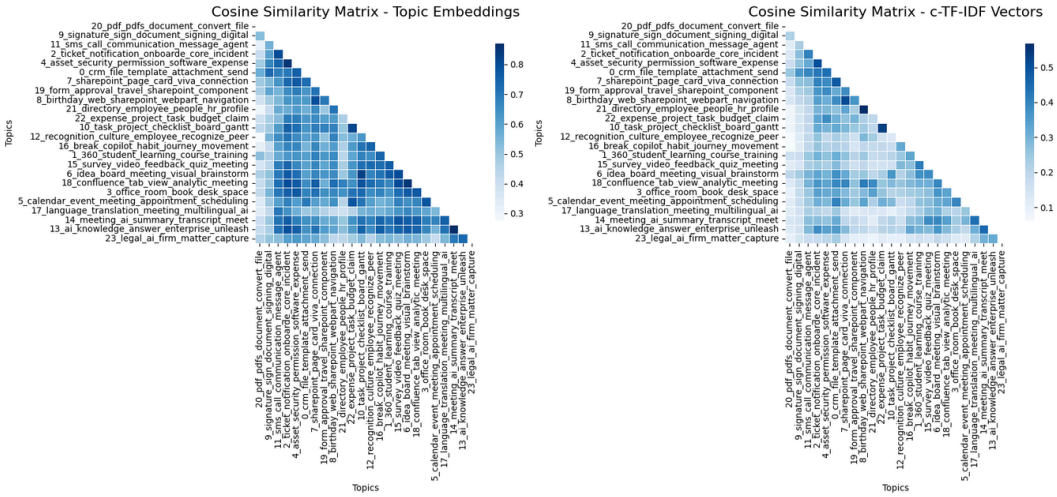


**Figure 6.14:** Heatmaps of topics semantic similarities based on Topic Embeddings (left) and c-TF-IDF Vectors (right)

Both heatmaps show similar patterns even if the values are different, with the c-TF-IDF similarities being generally lower than those from the Topic Embeddings. The Table 6.13 shows some examples of topics that are semantically similar, moderately similar and dissimilar to the topic 10 (task, project, checklist, board, gantt) for both the approaches.

| Topic | Embedding Sim. | c-TF-IDF Sim. |
|---|---|---|
| 22: (*expense*, *project*, *task*, *budget*, *claim*) | 0.81 | 0.54 |
| 19: (*form*, *approval*, *travel*, *sharepoint*, *component*) | 0.68 | 0.25 |
| 20: (*pdf*, *pdfs*, *document*, *convert*, *file*) | 0.43 | 0.08 |

**Table 6.13:** Examples of topic pairs with high, medium, and low semantic similarity to topic 10 (*task*, *project*, *checklist*, *board*, *gantt*)

The analysis of permissions similarities is performed by computing the pairwise cosine similarities between topics based on the average binary permissions vectors of the applications within each topic. The heatmap of permissions similarities in Figure 6.15 shows that most of the topics are quite similar in terms of permissions with some exceptions, suggesting the presence of recurring patterns of permissions across all applications in the dataset. This analysis highlights the need of a more granular analysis at the application level within each topic.

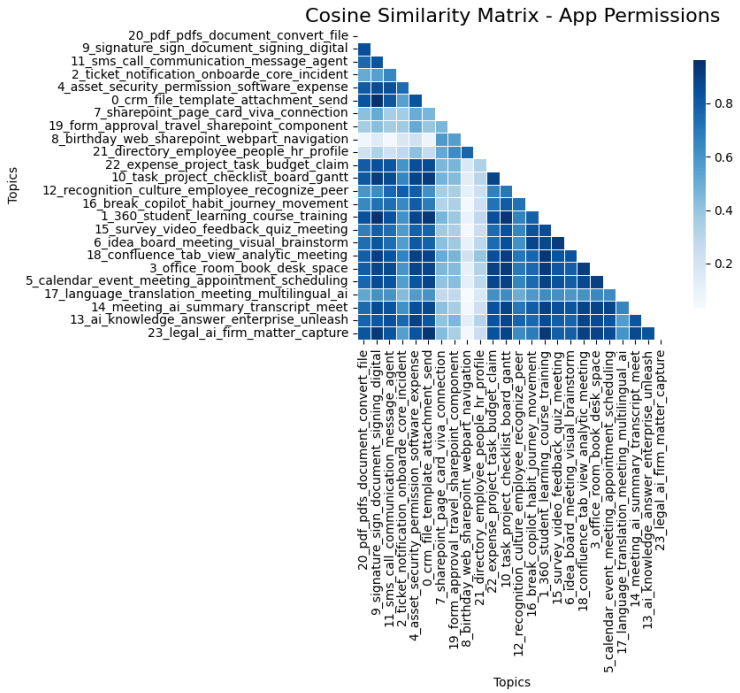


**Figure 6.15:** Heatmap of topics permissions similarities based on average binary permissions vectors

As we could expect, the permissions similarities do not perfectly align with the semantic similarities. This is mainly due to the presence of common permissions required by quite all applications, such as `User.Read` and `openid`, which tend to increase the overall similarity between topics. Furthermore, topic size can also impact on the pairwise cosine similarities since the binary permissions vectors are averaged across all applications in each topic before computing the pairwise cosine similarities. As a result, a permission required by a single application in a small topic can have

the same weight as a permission required by several applications in a large topic.

Despite that, we can still find some correlations intersecting the two approaches. Keeping the Topic 10 (task, project, checklist, board, gantt) as reference and the three examples in Table 6.13, we can see that the topic with high semantic similarity (expense, project, task, budget, claim) has also a quite high permissions similarity (0.88) and the topic with medium semantic similarity (form, approval, travel, sharepoint, component) has a medium permissions similarity (0.44). On the other hand, the topic with low semantic similarity (pdf, pdfs, document, convert, file) has a quite high permissions similarity (0.82). Comparing the top-10 permissions required by this pair of topics (i.e., Topic 10 and Topic 20), we can see that they share several user info and mail related permissions and their sizes are quite different (50 vs 15 applications), which can explain the high permissions similarity despite the low semantic similarity.

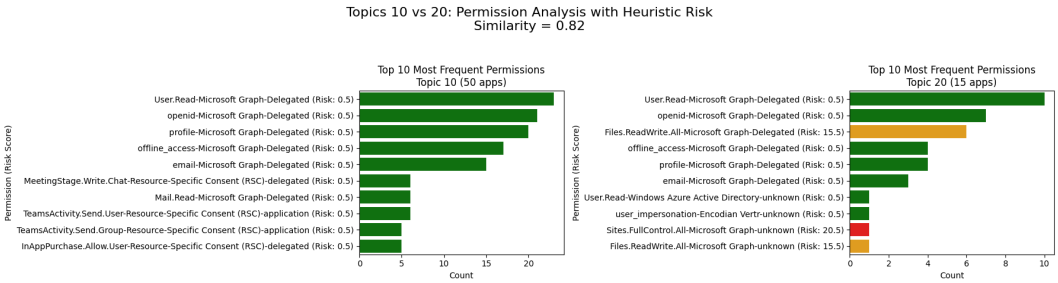


**Figure 6.16:** Topic 10 vs Topic 20: Top 10 permissions required by applications in each topic.

### Summary

The topic modelling analysis provided interesting insights about the nature of the applications in our dataset. The extracted topics effectively reflect the functionalities and the intents of the applications populating them, with a high degree of consistency between the themes of the topics and their most frequent permissions. Furthermore, the analysis of topics similarities showed that in some cases semantically related topics also share similar permissions patterns. However, the presence of common permissions and the few applications in our dataset limits the effectiveness of this analysis.

## 6.2 Anomaly Detection Results

This section presents the results obtained after applying the Anomaly Detection methodology introduced in Section 5.2.1. We start by analyzing the evaluation of the models on the injected outliers (supervised task), and then we focus on the identification of the native outliers (unsupervised task).

### 6.2.1 Building the Injected Outliers

As already mentioned, we used three distinct anomaly detection algorithms to identify outliers in each topic: One-Class SVM (OC-SVM), Isolation Forest (iForest), and Local Outlier Factor (LOF). This means that for each algorithm we train and evaluate a separate model for each topic. Before applying the models directly on the topics, we first want to assess their performance on identifying anomalies and select the best hyper-parameters for each algorithm. The complete lists of parameters for each strategy and their values for each level of difficulty are reported in Tables 6.14 and 6.15.

| **Difficulty** | **Parameters (Values)** |
|---|---|
| Easy | `min_threshold` = 0.0, `max_threshold` = 0.2 |
| Medium | `min_threshold` = 0.4, `max_threshold` = 0.6 |
| Hard | `min_threshold` = 0.8, `max_threshold` = 1.0 |

**Table 6.14:** Parameters and values for Cross-Topic injected outlier strategy at different difficulty levels.

| **Difficulty** | **Parameters (Values)** |
|---|---|
| Easy | `num_common_to_remove` = 1, `num_rare_to_add` = 1 |
| Medium | `num_common_to_remove` = 2, `num_rare_to_add` = 2 |
| Hard | `num_common_to_remove` = 4, `num_rare_to_add` = 4 |

**Table 6.15:** Parameters and values for Synthetic injected outlier strategy at different difficulty levels.

We evaluated the performance of each model on each injection strategy (e.g., Hard Synthetic) plotting the average ROC curve across all topics. On the x-axis of the ROC curve we report the False Positive Rate (FPR) which represents the percentage of native outliers detected by the model, while on the y-axis we report the True Positive Rate (TPR) which represents the percentage of injected outliers correctly flagged by the model. The objective is to maximize the TPR while keeping an acceptable FPR (i.e., not flagging too many native applications as outliers).

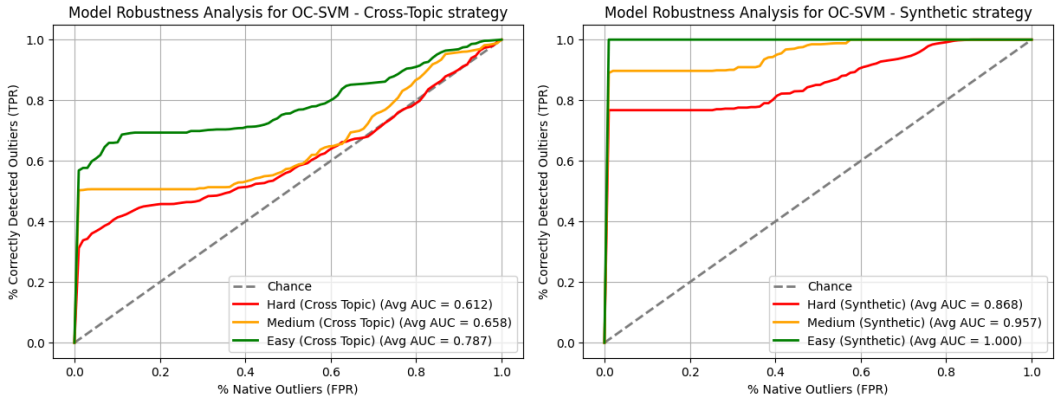


**Figure 6.17:** Average ROC curves for OC-SVM Cross-Topic (left) and Synthetic (right) injected outlier strategies across all topics.

As shown in the average ROC curves in Figure 6.17, the OC-SVM model perfectly reflects the difficulty levels of both strategies. The Cross-Topic strategy seems to be more challenging due to the presence of real applications as outliers that can be semantically closer to the applications in the topic under evaluation, making them harder to distinguish. On the other hand, the Synthetic strategy results to be easier with the easy and medium levels that are almost perfectly solved by the model.

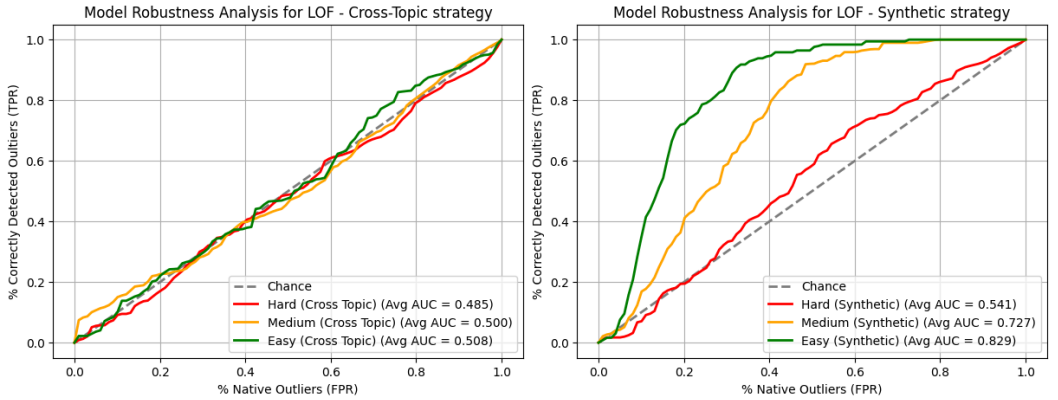


**Figure 6.18:** Average ROC curves for LOF Cross-Topic (left) and Synthetic (right) injected outlier strategies across all topics.

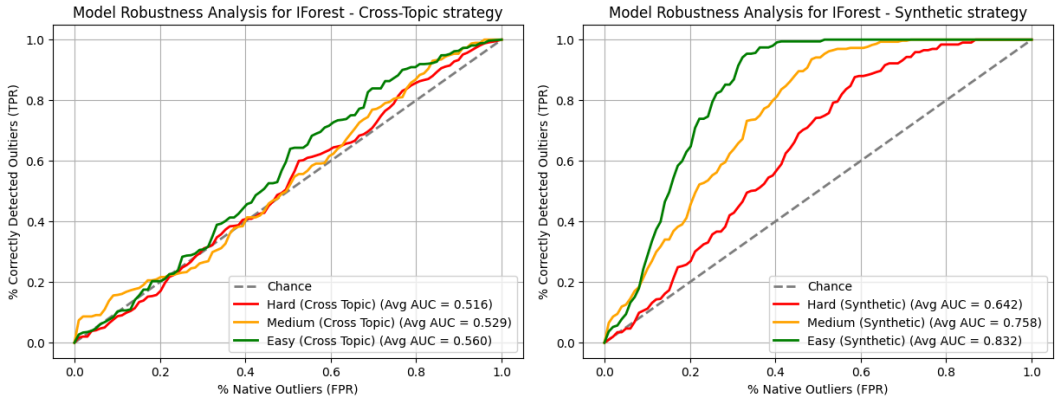


**Figure 6.19:** Average ROC curves for Isolation Forest Cross-Topic (left) and Synthetic (right) injected outlier strategies across all topics.

Both LOF and iForest models show difficulty in solving the Cross-Topic strategy, with the three levels almost overlapping in the ROC curve and achieving performance close to random guessing. On the other hand, the Synthetic strategy proved to be more manageable, with the easy level providing reasonable performance, while the medium and hard levels are more challenging.
Based on these results, we decided to use the Synthetic Outliers Injection strategy. The difficulty levels chosen for each algorithm are reported in Table 6.16 along with the best global hyperparameters identified after the grid search with k-fold cross-validation.

| **Algorithm** | **Selected Difficulty Level** | **Best Hyperparameters** |
|---|---|---|
| OC-SVM | Medium | `nu`=0.1 |
| LOF | Easy | `k`=3 |
| Isolation Forest | Easy | `n_estimators`=150 |

**Table 6.16:** Selected difficulty levels for each anomaly detection algorithm using Synthetic injected outlier strategy.

### 6.2.2 Model Performance on Injected Outliers

Once selected the best strategy for outlier injection and the best hyper-parameters for each algorithm, we can evaluate the performance of each model on the injected outliers. As already mentioned, now we evaluate the model from the perspective of the final task of native outlier identification. Each model with the optimal hyper-parameter is trained on the whole topic and evaluated on the outliers injected using the selected strategy and difficulty level. We visualize the results in terms of average ROC curve across all topics for each algorithm, allowing a direct comparison of the ability of each model to identify the injected outliers and the corresponding portion of native outliers flagged by the model (i.e., FPR). All the anomaly detection algorithms used in this project are designed to work with thresholds and anomaly scores. Specifically, OC-SVM has a threshold which is implicitly set to zero, as the

anomaly score is the distance from the separating hyperplane, while LOF and iForest require the setting of a threshold on the anomaly scores to classify an application as normal or outlier. For these two algorithms, we also provide an analysis of the optimal thresholds for each topic, selecting the best global threshold across all topics based on the median of optimal thresholds. The optimal threshold for each topic is identified by maximizing the Youden's Index, i.e., TPR - FPR.

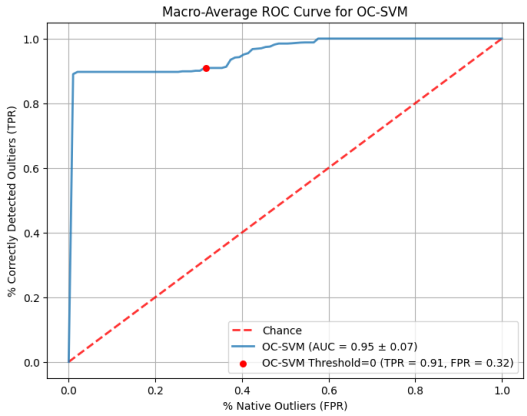


**Figure 6.20:** Average ROC curve for OC-SVM: Synthetic-Medium injected outlier strategy across all topics.

As already observed in the preliminary evaluation, the OC-SVM model performs quite well in identifying the injected outliers even on a Medium level task, achieving an average AUC score of 0.95 with low standard deviation across topics (0.07). The red scatter point in the ROC curve identifies the performance of the model when using the default threshold (i.e., 0) which results in a TPR of 0.91 (good ability to identify injected outliers) and FPR of 0.32 (i.e, on average 32% of applications are flagged as native outliers). These results are reasonable considering the difficulty of the task and the nature of our topics, with some of them being quite small.

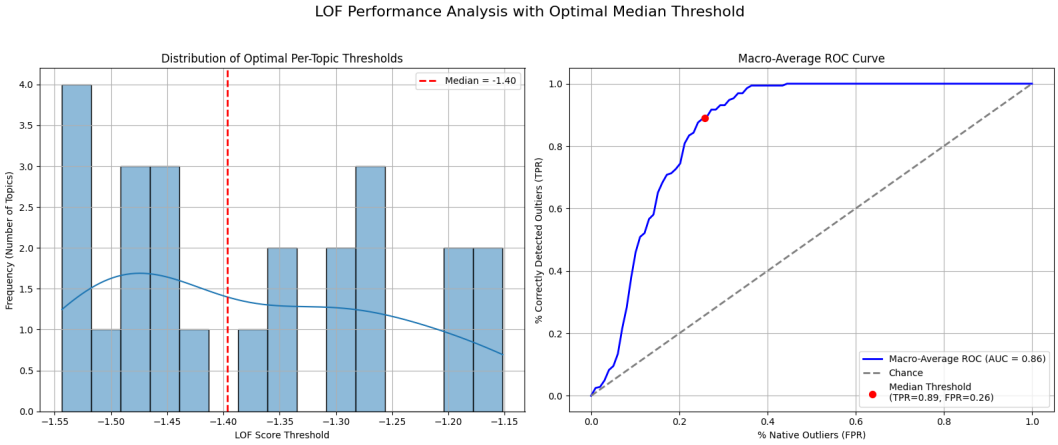


**Figure 6.21:** Average ROC curve for LOF and optimal thresholds distribution: Synthetic-Easy injected outlier strategy across all topics.

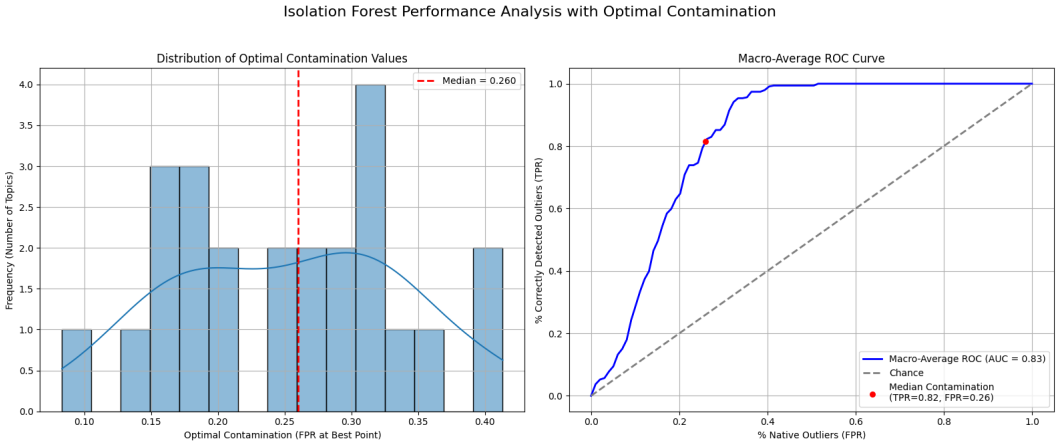


**Figure 6.22:** Average ROC curve for Isolation Forest and optimal thresholds distribution: Synthetic-Easy injected outlier strategy across all topics.

For LOF and iForest models, we perform the evaluation using the Easy level of the Synthetic strategy. In each Figure 6.21 and 6.22, we report the distribution of optimal thresholds across all topics along with the median value (on the right) and the average ROC curve (on the left). The LOF model achieves on average an AUC score of 0.86 while the iForest reaches 0.84 which are reasonable results considering the simplicity of the task and the difficulty that models faced with other strategies. The scatter red points on each ROC curve represent the performance of the models using the median optimal threshold across all topics (i.e., -1.40 of LOF score and 0.26 of contamination for iForest). At these thresholds, the LOF model achieves a TPR of 0.89 and FPR of 0.26, while the iForest model achieves a TPR of 0.82 and FPR of 0.26. These results are quite good reflecting a good balance between sensitivity and specificity.

The summary of the selected thresholds and the corresponding performance (TPR, FPR) is reported in Table 6.17.

| **Algorithm** | **Selected Threshold** | **TPR** | **FPR** |
|---|---|---|---|
| OC-SVM | 0 (default) | 0.91 | 0.32 |
| LOF | -1.40 (median) | 0.89 | 0.26 |
| Isolation Forest | 0.26 (median) | 0.82 | 0.26 |

**Table 6.17:** Summary of selected thresholds and corresponding performance for each anomaly detection algorithm on injected outliers.

### 6.2.3 Native Outlier Identification

In this section we provide the results of the native outlier identification task. Once determined the best models hyper-parameters and thresholds for each algorithm, we apply them to identify the native outliers in each topic. The percentage of flagged applications by each algorithm for each topic is reported in Figure 6.23.

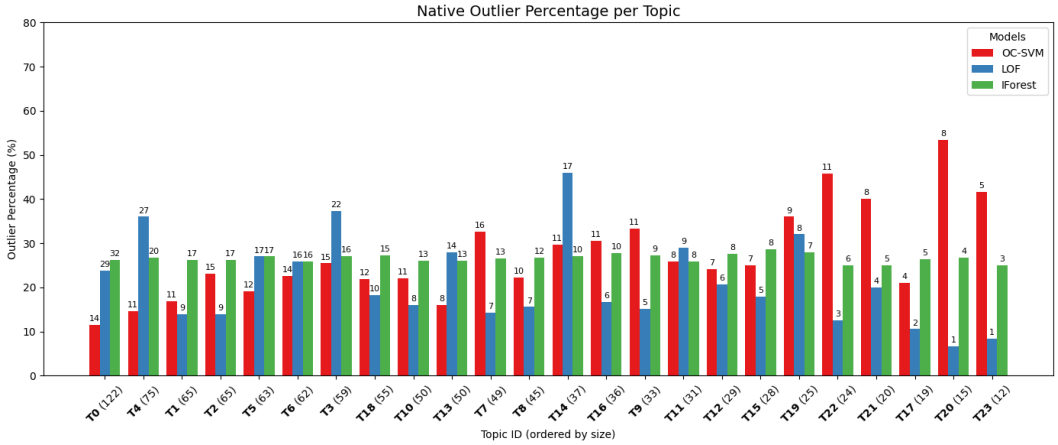


**Figure 6.23:** Native Outlier Percentage per Topic - Models Comparison

We can observe that the percentage of applications flagged in each topic is quite balanced, as most of the topics have between 20% and 30% of native outliers being reasonable for our case. However, some topics have a slightly higher percentage of flagged applications when using the OC-SVM model (e.g., Topic 20, with almost 50% of applications flagged as outliers) compared to the other two models. This is mainly due to the fact that LOF and iForest models were trained using the optimal thresholds making them more conservative in flagging applications and showing a more balanced behavior across topics. Another interesting consideration is that the topics with a small number of applications (e.g, Topic 23 with 12 applications) should be treated with caution as the models may not have enough support to learn a good representation of normality.

## 6.3 Evaluation of Findings

This section presents the evaluation of the findings obtained from the anomaly detection task. We start by comparing the results of the three algorithms used in this work with a topic-less baseline anomaly detection approach with OC-SVM. Then, we perform a cross-model analysis to identify common applications flagged as outliers by the different algorithms (both topic-based and topic-less). Finally, we validate our findings against the self-defined risk score introduced in Section 4.3.2 showing that the applications flagged as outliers tend to have a higher risk score compared to the normal applications.

### 6.3.1 Comparison with Topic-Less Baseline

To evaluate the effectiveness of our topic-based anomaly detection approach, we defined a baseline OC-SVM model trained of the whole dataset without considering the topics. The hyper-parameter `nu` is selected following the same procedure used for the topic-based OC-SVM models, using the Synthetic-Medium injected outlier strategy. The model achieves an average AUC score of 0.86 for both `nu=0.5` and

`nu=0.1`. We selected `nu=0.1` which is the same value used for the topic-based OC-SVM models. The average ROC curve of the baseline model is shown in Figure 6.24.

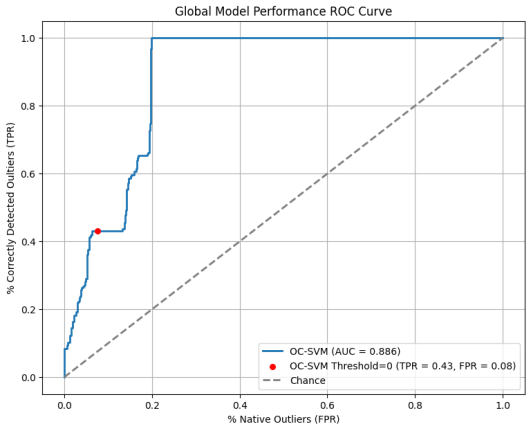


**Figure 6.24:** Average ROC curve for Topic-less OC-SVM: Synthetic-Medium injected outlier strategy across all topics.

Like the topic-based OC-SVM model, the red scatter point in the ROC curve identifies the performance of the model when using the default threshold (i.e., 0) which results in a TPR of 0.43 and FPR of 0.08. This performance is significantly lower than the topic-based approach, with a TPR of 0.91 and FPR of 0.32. This behavior is expected as the topic-less approach has to deal with a more heterogeneous dataset and injected outliers that might be quite similar to the general population of applications. The Topic-less OC-SVM model totally flags 99 applications as outliers, which is about the 10% of the whole dataset. The distribution of these applications across topics is shown in Figure 6.25.

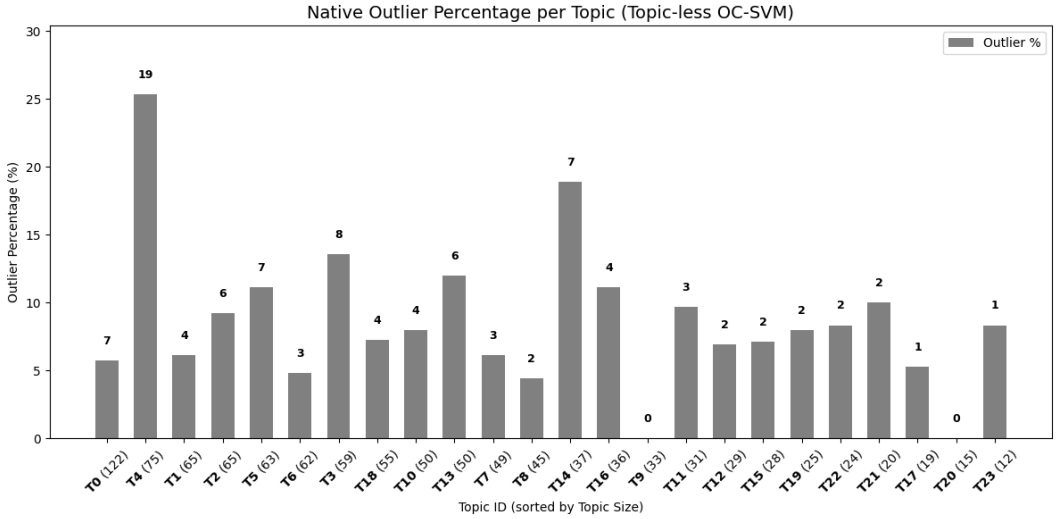


**Figure 6.25:** Distribution of native outliers identified by Topic-less OC-SVM across topics.

As we could expect, some topics have no applications flagged as outliers (i.e.,

Topic 9, Topic 20), while other topics have a significant number of applications flagged (e.g., Topic 4). This behavior reflects the topic-less approach used in this baseline model. Indeed, Topic 4 is related to "Security and Asset Management" which might contain applications whose permissions are uncommon when compared to the dataset as a whole, while Topic 9 and Topic 20 are respectively related to "Document Signatures" and "PDF Management" which typically have more standard permissions across all applications.

### 6.3.2 Cross-Model Analysis

At this point, we have a total of four sets of applications flagged as outliers: three sets from the topic-based models (OC-SVM, LOF, iForest) and one set from the topic-less OC-SVM model. We perform a cross-model analysis to understand the relationships between these sets and see if the models agree on some applications being outliers. The Figure 6.26 shows the portion of applications flagged as outliers by each model (on the left) and the pairwise model agreement matrix (on the right).

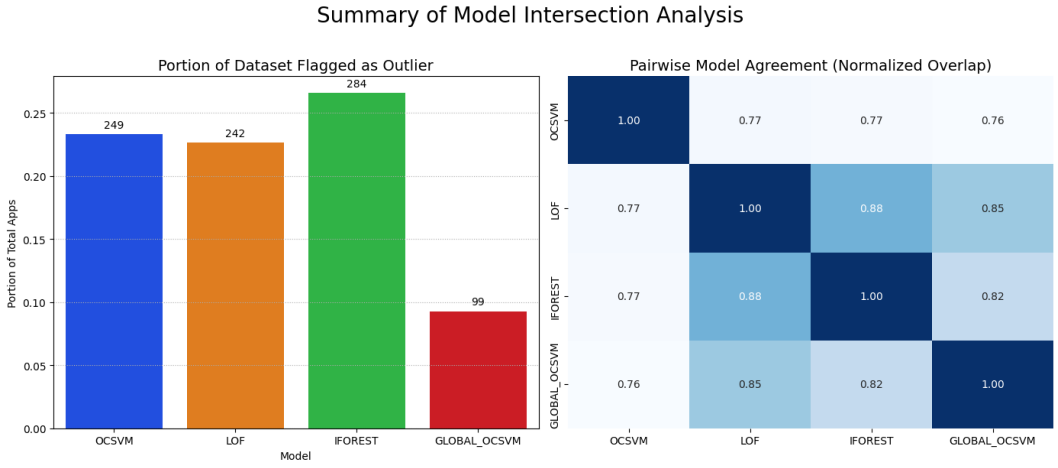


**Figure 6.26:** Portion of applications flagged as outliers by each model (left) and pairwise model agreement matrix (right).

iForest is the most sensitive model, flagging the 26% (284) of applications as outliers (as indicated by its contamination parameter set to 0.26), while OC-SVM and LOF show a similar behavior (22% of native outliers), flagging respectively 249 and 242 applications. As expected and already discussed, the topic-less OC-SVM model is the most conservative one due to the nature of the approach.

The pairwise model agreement matrix shows that LOF and iForest have the highest agreement among the topic-based models, with an 88% of applications categorized in the same way (either normal or outlier) by both models. The topic-based OC-SVM model has a moderate level of agreement with LOF and iForest (77% for both models). On the other hand, the topic-less OC-SVM model has an high level of agreement with LOF and iForest (85% and 82% respectively), while the agreement with the topic-based OC-SVM model is slightly lower (about 76% ).

This analysis is enforced by computing the Agreement Score defined in Section 5.3 for specific groups of models. The results are reported in Table 6.18.

| **Models** | **Agreement Score** |
|---|---|
| All Models (4) | 0.37 |
| Topic-based Models (3) | 0.44 |

**Table 6.18:** Agreement Scores for different groups of models.

The results suggest that topic-based models have a level of agreement of 0.44. This means that given the decision of one topic-based model (either inlier or outlier) there is a 44% of chance that all the other topic-based models will make the same decision. The agreement score slightly decreases (0.37) when considering all models, indicating that the topic-less OC-SVM Model has a different perspective on the data compared to the topic-based models, but still maintains a reasonable level of agreement.

### 6.3.3 Validation Against Self-Defined Risk Score

To validate our findings, we use the self-defined risk score introduced in Section 4.3.2. We compute the risk score for each application summing the risk score of its permissions. First, we verified if the applications flagged as outliers by the different models tend to have a higher risk score and if there are significant differences in the risk score distributions. Second we compared the risk score distributions of the outliers flagged by each model to check for significant differences. Finally we planned a manual review of some representative applications as case studies.

#### Risk Score Distributions

The Figure 6.27 shows the distributions of risk scores for all the applications populating our dataset. The histogram on the left shows that most of applications have low risk scores (between 0 and 50) with a long tail of applications with higher risk scores (up to about 370). Half of the applications have a risk score lower than 2.50 (median value), and less than 10% of applications have a risk score higher than 50, as shown in the ECDF on the right.

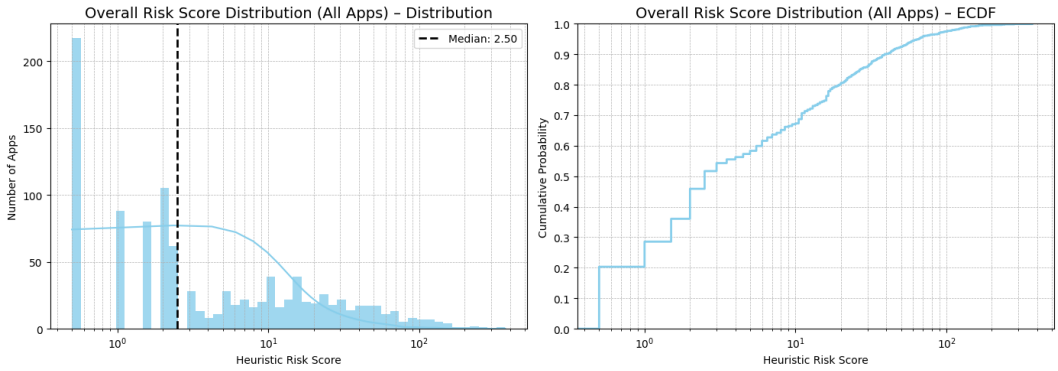


**Figure 6.27:** Distribution of risk scores for all applications in the dataset (left: histogram, right: ECDF).

Considering the applications which are not flagged as outliers by any model (656 applications) the risk score distribution significantly shifts to the left, with a median value of 1.50 and about the 10% of applications having a risk score higher than 10. As expected, the models collectively succeed in flagging most of the high-risk applications.

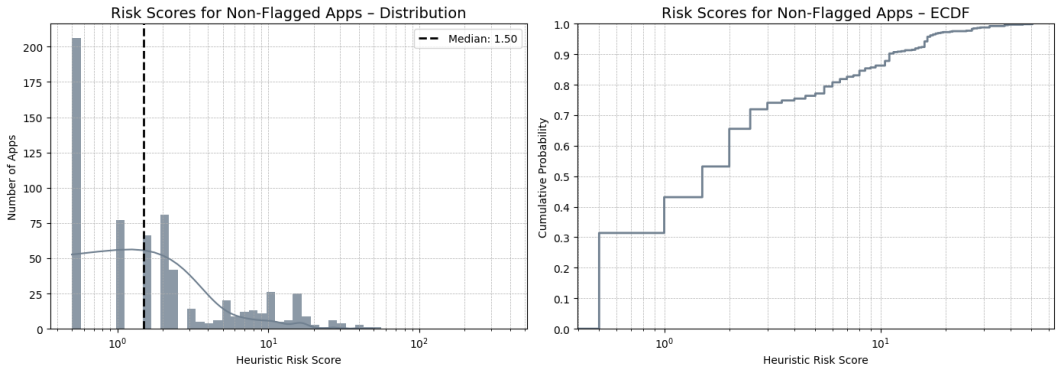


**Figure 6.28:** Distribution of risk scores for applications not flagged as outliers by any model (left: histogram, right: ECDF).

The histograms in Figure 6.29 show the risk score distributions for the applications both flagged (red) and not flagged (blue) as outliers by each model. The topic-less OC-SVM model tends to flag applications with higher risk scores compared to the topic-based models, with a median value of 58, as it tends to flag applications with uncommon permissions considering the whole dataset. On the other hand, the topic-based models still show a clear shift to the right, but with the tendency to flag more applications with low-to-medium risk scores. An interesting observation, is that the topic-based OC-SVM model left unflagged 13 high-risk applications (risk score $>$ 100), unlike LOF and iForest models which flag almost all of them.

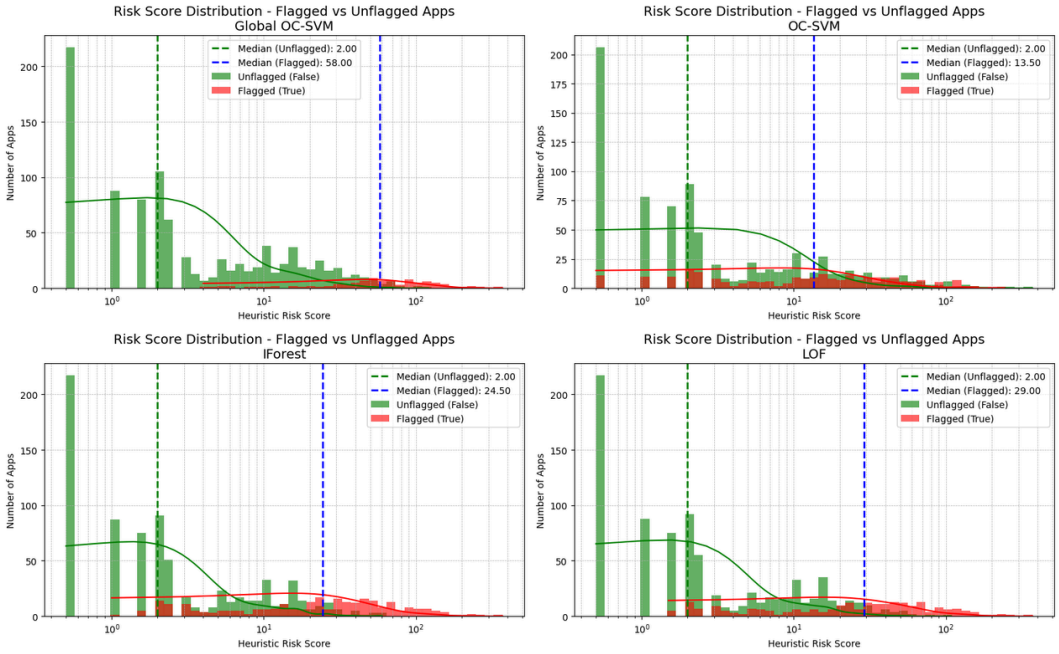


**Figure 6.29:** Distribution of risk scores for applications flagged (red) and not flagged (green) as outliers by each model (top-left: Topic-less OC-SVM, top-right: Topic-based OC-SVM, bottom-left: iForest, bottom-right: LOF).

## 6.4 Analysis of Model Predictions on Representative Applications

In this section we provide a qualitative analysis of the predicions made by the different models. The analysis is structured around three key groups of applications to highlight the strengths and weakenesses of the presented work: (1) applications flagged as outliers by all topic-based models but not by the topic-less model, (2) applications flagged as outliers by all topic-based models but not by the topic-less model, (3) high-risk applications not flagged by the topic-based OC-SVM model.

To provide a comprehensive overview of each case, we present a uniform format for the analysis. It includes the application's name, its assigned topic, a representative quote from its description, its risk score, the list of permissions it requires, the prediction of the models, and a final qualitative analysis of the case with a critical evaluation of the models' decisions.

### Topic-Based but not Topic-Less Flagged Applications

## Vacation Tracker

**Topic:** 3 (office, room, book, desk, space), **Topic Size**: 59
**Risk Score:** 43
**Qualitative Analysis: Suspicious**

**Description**

> *"Manage your team's PTO worry-free within Microsoft Teams, Outlook and Office."*

**Permissions** The application requests 13 permissions, including a critical permission with **Application scope**:

- **Application Permissions (acts on its own):**
  - `Calendars.ReadWrite`
- **Delegated Permissions (acts on behalf of a user):**
  - `Group.Read.All`, `User.Read.All`, `User.ReadBasic.All`
  - `MailboxSettings.ReadWrite`, `MailboxSettings.Read`
  - `Team.ReadBasic.All`, `User.Read`
  - `email`, `openid`, `profile`, `offline_access`
  - One Resource-Specific Consent (RSC) permission.

**Model Predictions**

- **OC-SVM (Topic-Based):** Flagged
- **LOF (Topic-Based):** Flagged
- **iForest (Topic-Based):** Flagged
- **OC-SVM (Topic-Less):** Not Flagged

**Analysis** The application is a vacation tracker which allows users to manage their paid time off (PTO) within Microsoft Teams, Outlook, and Office. It was correctly assigned to Topic 3, which focuses on bookings and office space management services. However, the application requests permissions that are excessive and not justified by its described functionality. The most concerning permissions is `Calendars.ReadWrite` with **Application scope**, which allows the application to read, create, update, and delete events in every user's calendars across the entire tenant. Also permissions such as `MailboxSettings.ReadWrite`

and `MailboxSettings.Read` with delegated scope are not necessary for a simple vacation tracker.

This is a perfect example of the advantage of using a topic-based approach for anomaly detection, as the topic-less model likely viewed the over-privileged permissions as normal across the entire dataset.

## Seismic for Outlook

**Topic:** 0 (crm, file, template, attachment, send), **Topic Size**: 122
**Risk Score:** 30
**Qualitative Analysis: Suspicious**

**Description**

> *"Seismic for Outlook empowers Marketing and Sales teams by putting Seismic content and meeting workflows directly where sellers work — in email and on the calendar...Sellers can create, link, and manage Seismic Meeting drafts without leaving Outlook...Quickly search for and share Seismic content, manage agendas, attach content, and launch presentations seamlessly from Outlook."*

**Permissions** The application requests 10 permissions, all of which are **Delegated**:

- `Directory.Read.All`
- `Group.Read.All`, `GroupMember.Read.All`
- `Contacts.Read`, `Contacts.Read.Shared`
- `User.Read`, `User.ReadBasic.All`
- `email`, `openid`, `profile`

**Model Predictions**

- **OC-SVM (Topic-Based):** Flagged
- **LOF (Topic-Based):** Flagged
- **iForest (Topic-Based):** Flagged
- **OC-SVM (Topic-Less):** Not Flagged

**Analysis** The application is a sales management tool integrated with Outlook that helps in content sharing and meeting workflows. It is correctly assigned to Topic 0 which is related to templates and attachment management tools. The disconnection here arises from the application requesting the `Directory.Read.All` permission which grants the ability to read the entire organization's directory structure. This need of this level of access is not justified by the stated functionality of the application. The context provided by the topic helps in identifying this as anomaly, unlike the topic-less model, showing the advantage of a topic-based approach.

## TeamOrgChart

**Topic:** 8 (birthday, web, sharepoint, webpart, navigation), **Topic Size**: 45
**Risk Score:** 3
**Qualitative Analysis: Suspicious (Functional Anomaly)**

**Description**

> *"TeamOrgChart+ is an easy-to-configure organization chart solution that's highly secure and synchronized with Microsoft Teams, Outlook & Office...Swift to deploy, TeamOrgChart+ increases awareness, supports re-organizations and assists with staff onboarding...*
> *TeamOrgChart+ can use a range of data sources including Active Directory, SQL databases, SharePoint and Excel to create your organization charts."*

**Permissions** The application requests only 5 minimal, **Delegated** sign-in permissions:

- `User.Read`, `email`, `openid`, `profile`, `offline_access`

**Model Predictions**

- **OC-SVM (Topic-Based):** Flagged
- **LOF (Topic-Based):** Flagged
- **iForest (Topic-Based):** Flagged
- **OC-SVM (Topic-Less):** Not Flagged

**Analysis** This is a True Positive in terms of being an unusual application, but not in the traditional sense of over-privileged permissions. The application

is designed to generate and manage organizational charts and it is correctly assigned to Topic 8, which focuses on SharePoint web parts and related tools. The anomaly here is a suspicious *lack* of necessary permissions. The app only requests minimal sign-in permissions and does not request any permissions for its core functionalities, which likely are prompted during use (this case is not handled by our data collection pipeline). The topic-based models likely flag this types of applications because they significantly deviate from the permissions patterns within topics in our dataset. On the other hand, the topic-less model does not flag it because more applications in the whole dataset follow this minimal permission pattern.

### Applications Flagged by All Models

## monday.com

**Topic:** 10 (task, project, checklist, board, gantt), **Topic Size**: 50
**Risk Score:** 126
**Qualitative Analysis: Suspicious**

**Description Quotes**

> *"monday.com is a versatile Work OS that powers remote teamwork...Through monday.com, you can use building blocks – such as boards, views, charts, automations, and integrations – to create custom workflow apps to run processes, projects and everyday work...monday.com connects your team so you can continue to collaborate, manage and track work in one easy-to-use platform."*

**Permissions** The application requests 14 permissions in total, grouped by type below.

- **Application Permissions (acts on its own):**
  - `TeamsAppInstallation.ReadWriteSelfForUser.All`
  - `User.Read.All`
- **Delegated Permissions (acts on behalf of a user):**
  - `Application.Read.All`, `Application.ReadWrite.All`
  - `User.Export.All`, `User.Invite.All`
  - `User.ManageIdentities.All`
  - `User.Read`, `User.Read.All`, `User.ReadBasic.All`

- `User.ReadWrite`, `User.ReadWrite.All`
- `UserAuthenticationMethod.Read.All`
- `UserAuthenticationMethod.ReadWrite.All`

**Model Predictions**

- **OC-SVM (Topic-Based):** Flagged
- **LOF (Topic-Based):** Flagged
- **iForest (Topic-Based):** Flagged
- **OC-SVM (Topic-Less):** Flagged

**Analysis** The application is designed to make easy the management of team projects, providing a platform for collaboration where tasks, team members, and custom workflows can be organized and tracked. It is correcly assigned to Topic 10, which is related to project and task management tools. However, the application requests permissions to read and modify the authentication methods of all users in the organization (`UserAuthenticationMethod.Read.All` and `UserAuthenticationMethod.ReadWrite.All`) on behalf of the signed-in user. The description does not justify the need for such permissions, and applications in this category do not require such high privileges.

## Officely - Desk Booking

**Topic:** 3 (office, room, book, desk, space), **Topic Size**: 59
**Risk Score:** 116
**Qualitative Analysis: Suspicious**

**Description Quotes**

> *"Officely allows your team to see who will be in the office, book in alongside them, reserve a car parking space and book a meeting room, directly in Microsoft Teams...Need a desk? A meeting room? A parking spot? Lunch? Even your dog's spot? You can book it all right from Microsoft Teams...Unlike other desk booking tools, Officely lives right in Microsoft Teams. No extra apps or logins to remember, just higher adoption and a happy team."*

**Permissions** The application requests 14 permissions in total, grouped by type below.

- **Application Permissions (Microsoft Graph):**
  - `AppCatalog.Read.All`
  - `Chat.Create`, `ChatMember.ReadWrite.All`, `Chat.ManageDeletion.All`
  - `Channel.ReadBasic.All`, `Team.ReadBasic.All`
  - `Organization.Read.All`, `User.Read.All`
  - `TeamsAppInstallation.ReadWriteSelfForUser.All`
  - `TeamsAppInstallation.ReadWriteAndConsentSelfForTeam.All`
- **Application Permissions (Resource-Specific Consent):**
  - `Member.Read.Group`
  - `TeamSettings.Read.Group`
  - `ChannelSettings.Read.Group`
- **Delegated Permissions (acts on behalf of a user):**
  - `User.Read`

**Model Predictions**

- **OC-SVM (Topic-Based):** Flagged
- **LOF (Topic-Based):** Flagged
- **iForest (Topic-Based):** Flagged
- **OC-SVM (Topic-Less):** Flagged

**Analysis** The application is a desk and workspace booking tool integrated within Microsoft Teams. It was correctly assigned to Topic 3, which focuses on office and room booking solutions. Despite some permissions being justified (e.g., `User.Read.All` to identify users booking desks), the overall permission set is excessive considering the application's stated purpose in the description. Permissions like `ChatMember.ReadWrite.All` and `Chat.ManageDeletion.All` are not only unnecessary but represent a severe security risk, as they allow the application to add or remove members from any chat and delete any chat message across the entire organization.

## Brochesia

**Topic:** 1 (360, student, learning, course, training), **Topic Size**: 65

**Risk Score:** 4
**Qualitative Analysis: Not Suspicious**

**Description Quotes**

> *"Brochesia is the app that makes the work of operators from the field smarter...Thanks to the use of Augmented Reality, our solution connects you with people geographically widespread and allows you to share live content and information using Smart Glasses or mobile devices...Anyone can activate an audio-video streaming session and share their point of view with the remote operator, overlaying information from the field thanks to Augmented Reality."*

**Permissions** The application requests 8 permissions, all of which are **Resource-Specific Consent (RSC)** with **Application scope**:

- `Channel.Create.Group`, `Channel.Delete.Group`
- `TeamsTab.Read.Group`, `TeamsTab.Create.Group`, `TeamsTab.Delete.Group`
- `ChannelMessage.Read.Group`, `ChannelSettings.Read.Group`, `TeamSettings.Read.Group`

**Model Predictions**

- **OC-SVM (Topic-Based):** Flagged
- **LOF (Topic-Based):** Flagged
- **iForest (Topic-Based):** Flagged
- **OC-SVM (Topic-Less):** Flagged

**Analysis** This application is designed to help on site workers to solve technical problems via Teams using Augmented Reality (if supported). It was correctly assigned to Topic 1, which focuses on learning and training solutions, even if it is not a classic learning platform. All the permissions are related to managing channels and tabs within a specific Team, which is aligned with the intents of improving collaboration and training. The permissions set is quite unusual, as it includes only RSC application permissions whose statistical occurence in the topic is quite uncommon. This highlights a limitation of the models, which may flag uncommon patterns of permissions if not enough represented in the topics (or dataset in general), stressing the need for a richer and more diverse dataset.

### High-Risk Applications Not Flagged by Topic-Based OC-SVM

## CSC Voice AI

**Topic:** 17 (language, translation, meeting, multilingual, ai), **Topic Size**: 18
**Risk Score:** 110
**Qualitative Analysis: Suspicious**

**Description Quotes**

> *"Real-time voice translation & transcription powered by Cloud Solutions Company."*

**Permissions** The application requests 26 permissions, including a mix of Application and very high-privilege Delegated scopes.

- **Key Anomalous Permission:** `User.ReadWrite.All` (Delegated)
- **Other Application Permissions:** `OnlineMeetings.ReadWrite.All`, `Calendars.Read.All`, `User.Read.All`
- **Other Delegated Permissions:** `Calendars.ReadWrite`, `User.ReadWrite`, `OnlineMeetings.ReadWrite`

**Model Predictions**

- **OC-SVM (Topic-Based):** Not Flagged
- **LOF (Topic-Based):** Flagged
- **iForest (Topic-Based):** Flagged
- **OC-SVM (Topic-Less):** Flagged

**Analysis** The application is designed as a real-time voice translation and transcription tool, and it is correctly assigned to Topic 17, which focuses on language and meeting-related tools. However, it requests the extremely high-risk permission `User.ReadWrite.All` with **Delegated scope**, which is not justified by its intended functionality. This makes the application a **False Negative** for the topic-based OC-SVM model. The permission allows any app user to read and modify the profile information of every user in the organization, including administrators, posing a significant security risk. Topic 17 is one of the smallest topics and this limits the ability of learning a good representation of normality for that topic. Algorithms like LOF and iForest, which are more sensitive to density variations, were able to detect this anomaly even in a small topic, unlike OC-SVM which struggled.

## 6.5 Summary

This chapter presented a comprehensive evaluation of the results obtained from the methodologies applied to a dataset of Microsoft 365 applications. First, we detailed the topic modelling process, including systematic hyper-parameter tuning of the BERTopic pipeline for extracting the most coherent topics from the dataset. The selected configuration resulted in a significant improvement with respect to the baseline configuration leading to the extraction of 24 interpretable topics which reflect the nature of the applications populating the dataset.

Subsequently, we evaluated three anomaly detection algorithms (OC-SVM, LOF, Isolation Forest). First we compared two outlier injection strategies (synthetic and cross-topic) for model selection and evaluation with the synthetic strategy resulted more effective. Then we applied the best configuration of each algorithm to directly identify anomalous applications within each topic as "native outliers" A topic-less OC-SVM model was also trained to provide a baseline for comparison.

A cross model analysis revealed a moderate agreement among models, with topic-based approaches identifying context-specific anomalies that the topic-less model often missed. The topic-based models outperformed the topic-less baseline, being able to capture stronger context to identify more suspicious apps. Validation against a self-defined risk score confirmed that flagged applications generally had higer risk.

Finally, a qualitative review of representative cases demonstrated the strengths and limitations of the models, emphasizing the value of topic context for detecting suspicious applications and the limitation of the models when not enough data is available to learn a good representation of normality. The results set the stage for future work and refinements, which will be discussed in the next chapter.

# Chapter 7

# Conclusion

As the shift of enterprises towards cloud-based productivity suites continues to accelerate, with M365 being a leading choice, the security of these ecosystems becomes a critical concern. The key role of applications in enhancing productivity and collaboration within M365 tenants is evident as they can offer a wide range of functionalities that extend the capabilities of the core services. However, the extensive permissions often required by these applications can lead to over-privilege, posing significant security and privacy risks and representing a possible attack vector for malicious actors. This thesis addressed the following main research question:

*"Is it possible to automatically identify overprivileged applications in the Microsoft 365 ecosystem?"*

To answer this question, we proposed an approach that leverages Neural Topic Modelling techniques to extract the intents of applications living in the M365 ecosystem and leverage this context to identify over-privileged applications through a mixed unsupervised-supervised anomaly detection pipeline.

Before applying the approach, we collected a dataset of applications working in M365, with their textual descriptions and requested permissions, merging data from multiple sources. Then we extracted 24 distinct topics from the dataset using BERTopic, achieving reasonable coherence and diversity scores. The topics showed a good level of interpretability and reflected the nature of applications populating M365 ecosystems, with most of the topics focusing on productivity and collaboration. For evaluating topics similarities, we leveraged cosine similarity computed on semantic embeddings and permissions features. The results showed that in some cases topics with similar intents also share similar permissions patterns. However the presence of recurring permissions and the low size of some topics limited the effectiveness of this analysis, showing not clear correlations. Once the topics were extracted, we used the captured semantic context to implement a pipeline for identifying over-privileged applications. We evaluated the performance of three different algorithms (OC-SVM, LOF, IForest) in identifying injected anomalies and then we used the best performing models to identify native outliers within topics. Collectively, the three algorithms showed a good level of agreement in taking decision about an

application being anomalous or not. The comparison with a topic-less OC-SVM model showed that introducing semantic context is effective in detecting anomalies that would be otherwise missed. The qualitative analysis of the detected outliers revealed that several applications flagged as anomalous requested permissions that were not coherent with their described intents, suggesting potential overprivilege.

## 7.1 Limitations

This study has several limitations that should be addressed for future research. The fragmented nature of the M365 ecosystem with multiple application sources, not easily accessible and the lack of a complete centralized repository for collecting application descriptions and permissions, posed a significant challenge for the data collection process.

This led to extracted data being likely not representative of the entire M365 ecosystem, limiting the effectiveness of the topic extraction and the implemented anomaly detection models. Additionally, the data cleaning and merging process may have introduced some inaccuracies and inconsistencies that could have affected the quality of the results, mainly due to the lack of a unique identifier for applications across sources.

Moreover, the unlabelled nature of the dataset made the evaluation of the implemented models particularly challenging, restricting it to the detection of injected anomalies and a qualitative analysis of the results.

## 7.2 Future Work

Future works could address the limitations identified in this work and expand the research in several directions:

- **Dataset Enhancement**: collecting a more representative and bigger dataset of applications working in M365 surely would improve the reliability of the results, providing a more comprehensive view of the ecosystem,
- **Ground Truth Labels**: having verified ground truth labels for at least a part of the dataset or developing an automated or semi-automated robust labelling method would enable for a more reliable evaluation of the implemented models,
- **Adding Features**: Exploring additional features, such as permissions usage patterns, and verified risk scores could further enhance the detection of over-privileged applications,
- **Explainable AI**: Explainable AI techniques could be used at the end of the pipeline to provide insights on why an application is flagged as anomalous and which specific permissions raise the suspicion,

# Bibliography


[1] Google Cloud. *Advantages of Cloud Computing*. 2025. URL: cloud.google.com/learn/advantages-of-cloud-computing (visited on 06/10/2024) (cit. on p. 1).

[2] Electro IQ. *Microsoft 365 Statistics by Downloads, Subscribers, Revenue and Facts*. Feb. 2025. URL: electroiq.com/stats/microsoft-365-statistics/ (visited on 06/10/2024) (cit. on p. 1).

[3] Wikipedia. *Pacchetto Software Prodotto Da Microsoft*. Apr. 2012. URL: it.wikipedia.org/wiki/Microsoft_365 (visited on 10/07/2025) (cit. on p. 1).

[4] Dev4Side. *What Is Microsoft 365 and How Does It Work*. 2020. URL: www.dev4side.com/en/blog/what-is-microsoft-365 (visited on 06/10/2024) (cit. on pp. 1, 4).

[5] SkyTerra Technologies. *The Benefits of Migrating to Office 365 - SkyTerra*. Aug. 2025. URL: skyterratech.com/migrating-to-office-365/ (visited on 09/16/2025) (cit. on p. 1).

[6] MSP Corp. *8 Key Benefits of Microsoft 365 for Your Business*. Feb. 2024. URL: mspcorp.ca/blog/8-key-benefits-of-microsoft-365/ (visited on 06/10/2024) (cit. on p. 1).

[7] Microsoft Learn. *Tenant Management for Microsoft 365 for Enterprise*. Sept. 2024. URL: learn.microsoft.com/en-us/microsoft-365/solutions/tenant-management-overview?view=o365-worldwide (visited on 09/16/2024) (cit. on pp. 1, 5).

[8] SentinelOne. *All Eyes on Cloud | Why the Cloud Surface Attracts Attacks*. Oct. 2022. URL: www.sentinelone.com/blog/all-eyes-on-cloud-why-the-cloud-surface-attracts-attacks/ (visited on 06/10/2024) (cit. on p. 2).

[9] Nick Martin. *Practical Strategies for Securing Cloud Data and Microsoft 365*. Feb. 2025. URL: www.forbes.com/councils/forbesbusinesscouncil/2025/02/03/practical-strategies-for-securing-cloud-data-and-microsoft-365/ (visited on 06/10/2024) (cit. on p. 2).

[10] AdminDroid. *Review App Permissions & Consents in Microsoft 365*. Oct. 2022. URL: blog.admindroid.com/review-app-permissions-consents-in-microsoft-365/ (visited on 09/17/2025) (cit. on p. 2).

[11] Push Security. *What's the Difference between Application and Delegated OAuth Permissions on Microsoft 365?* 2025. URL: pushsecurity.com/help/application-and-delegated-oauth-permissions (visited on 06/10/2024) (cit. on pp. 2, 7).

[12] Wikipedia. *Principle of Least Privilege*. July 2019. URL: en.wikipedia.org/wiki/Principle_of_least_privilege (visited on 06/10/2024) (cit. on p. 2).

[13] One.com. *What Is Office 365 and How Does Office 365 Work? Read It All Here*. URL: www.one.com/en/microsoft-office-365/what-is-office-365 (visited on 06/10/2024) (cit. on p. 4).

[14] Wikipedia. *Microsoft 365*. Oct. 2020. URL: en.wikipedia.org/wiki/Microsoft_365 (visited on 06/10/2024) (cit. on p. 4).

[15] Valence Security. *Microsoft 365 Security Explained: Best Practices, Features, and Assessment*. 2025. URL: www.valencesecurity.com/saas-security-terms/microsoft-365-security-explained-best-practices-features-and-assessment (visited on 08/19/2025) (cit. on p. 4).

[16] Microsoft Learn. *What Is Microsoft Entra?* July 2024. URL: learn.microsoft.com/en-us/entra/fundamentals/what-is-entra (visited on 06/10/2024) (cit. on p. 5).

[17] Quest. *What Is Microsoft Entra ID?* URL: www.quest.com/learn/what-is-microsoft-entra-id.aspx (visited on 06/10/2024) (cit. on p. 5).

[18] Microsoft Learn. *Overview of Microsoft Entra Role-Based Access Control (RBAC)*. Oct. 2023. URL: learn.microsoft.com/en-us/entra/identity/role-based-access-control/custom-overview (visited on 06/10/2024) (cit. on p. 5).

[19] Microsoft Learn. *Understand the Microsoft Entra Schema and Custom Expressions - Microsoft Entra ID*. Apr. 2025. URL: learn.microsoft.com/en-us/entra/identity/hybrid/cloud-sync/concept-attributes (visited on 09/18/2025) (cit. on p. 5).

[20] Microsoft Learn. *Learn about Groups and Group Membership - Microsoft Entra*. Sept. 2024. URL: learn.microsoft.com/en-us/entra/fundamentals/concept-learn-about-groups (visited on 06/10/2024) (cit. on p. 5).

[21] Microsoft Learn. *What Is Device Identity in Microsoft Entra ID?* Oct. 2023. URL: learn.microsoft.com/en-us/entra/identity/devices/overview (visited on 06/10/2024) (cit. on p. 5).

[22] Auth0. *What Is Authentication? Definition and Uses*. URL: auth0.com/intro-to-iam/what-is-authentication (visited on 06/10/2024) (cit. on p. 6).

[23] Securiti. *What Is Authorization?* Feb. 2025. URL: securiti.ai/glossary/authorization/ (visited on 06/10/2024) (cit. on p. 6).

[24] Michael Nieles et al. *An Introduction to Information Security*. Vol. 1. 1. June 2017. DOI: 10.6028/nist.sp.800-12r1. URL: nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-12r1.pdf (visited on 06/10/2024) (cit. on p. 6).

[25] Microsoft Learn. *OAUTH 2.0 Authorization with Microsoft Entra ID*. Oct. 2023. URL: learn.microsoft.com/en-us/entra/architecture/auth-oauth2 (visited on 06/10/2024) (cit. on p. 6).

[26] Microsoft Learn. *Apps and Service Principals in Microsoft Entra ID*. Oct. 2023. URL: learn.microsoft.com/en-us/entra/identity-platform/app-objects-and-service-principals?tabs=browser (visited on 06/10/2024) (cit. on p. 6).

[27] Dick Hardt. *RFC 6749: The OAuth 2.0 Authorization Framework*. 2025. URL: datatracker.ietf.org/doc/html/rfc6749#section-3.3 (visited on 06/10/2024) (cit. on p. 7).

[28] Microsoft Learn. *Overview of Microsoft Graph Permissions - Microsoft Graph*. Oct. 2023. URL: learn.microsoft.com/en-us/graph/permissions-overview?tabs=http (visited on 06/10/2024) (cit. on pp. 7, 8).

[29] Microsoft Learn. *Resource-Specific Consent for Apps - Teams*. Feb. 2025. URL: learn.microsoft.com/en-us/microsoftteams/platform/graph-api/rsc/resource-specific-consent (visited on 09/22/2025) (cit. on pp. 8, 24).

[30] Microsoft. *Microsoft Identity Platform Glossary - Microsoft Identity Platform*. May 2025. URL: learn.microsoft.com/en-us/entra/identity-platform/developer-glossary (visited on 09/20/2025) (cit. on p. 8).

[31] Gary Archer. *Scope Best Practices*. Sept. 2024. URL: curity.io/resources/learn/scope-best-practices/ (visited on 06/10/2024) (cit. on p. 8).

[32] Okta. *What Is the OAuth 2.0 Authorization Code Grant Type?* URL: developer.okta.com/blog/2018/04/10/oauth-authorization-code-grant-type (visited on 06/10/2024) (cit. on p. 9).

[33] Palo Alto Networks. *What Is the Principle of Least Privilege?* 2024. URL: www.paloaltonetworks.com/cyberpedia/what-is-the-principle-of-least-privilege (visited on 06/10/2024) (cit. on p. 10).

[34] NIST. *Least Privilege - Glossary | CSRC*. 2024. URL: csrc.nist.gov/glossary/term/least_privilege (visited on 06/10/2024) (cit. on p. 10).

[35] Microsoft Learn. *Best Practices for Using Microsoft Graph Permissions - Microsoft Graph*. Nov. 2024. URL: learn.microsoft.com/en-us/graph/best-practices-graph-permission (visited on 09/21/2025) (cit. on p. 10).

[36] Microsoft Learn. *Reduce Overprivileged Permissions and Apps*. May 2024. URL: learn.microsoft.com/en-us/security/zero-trust/develop/overprivileged-permissions (visited on 06/10/2024) (cit. on pp. 11, 13).

[37] AlteredSecurity. *Introduction to 365-Stealer - Understanding and Executing the Illicit Consent Grant Attack*. May 2021. URL: www.alteredsecurity.com/post/introduction-to-365-stealer (visited on 09/21/2025) (cit. on p. 12).

[38] Microsoft Learn. *Configure the Admin Consent Workflow - Microsoft Entra ID*. Apr. 2025. URL: learn.microsoft.com/en-us/entra/identity/enterprise-apps/configure-admin-consent-workflow (visited on 09/21/2025) (cit. on p. 14).

[39] Microsoft Learn. *Panoramica - Microsoft Defender for Cloud Apps*. Nov. 2024. URL: learn.microsoft.com/it-it/defender-cloud-apps/what-is-defender-for-cloud-apps (visited on 09/21/2025) (cit. on p. 14).

[40] Cloud-Architekt. *AzureAD-Attack-Defense/ConsentGrant.md at Main · Cloud-Architekt/AzureAD-Attack-Defense*. 2020. URL: github.com/Cloud-Architekt/AzureAD-Attack-Defense/blob/main/ConsentGrant.md#mitre-attck-framework (visited on 10/07/2025) (cit. on pp. 14, 15).

[41] Microsoft Learn. *Investigate App Governance Threat Detection Alerts - Microsoft Defender for Cloud Apps*. Aug. 2025. URL: learn.microsoft.com/en-us/defender-cloud-apps/app-governance-anomaly-detection-alerts#persistence-alerts (visited on 09/25/2025) (cit. on p. 15).

[42] Microsoft Learn. *Learn about Data Loss Prevention*. Sept. 2023. URL: learn.microsoft.com/en-us/purview/dlp-learn-about-dlp (visited on 06/10/2024) (cit. on p. 16).

[43] Microsoft Learn. *What Is Microsoft Sentinel?* May 2024. URL: learn.microsoft.com/en-us/azure/sentinel/overview?tabs=defender-portal (visited on 06/10/2024) (cit. on p. 16).

[44] Cloud-Architekt. *GitHub - Cloud-Architekt/AzureAD-Attack-Defense: This Publication Is a Collection of Various Common Attack Scenarios on Microsoft Entra ID (Formerly Known as Azure Active Directory) and How They Can Be Mitigated or Detected*. 2020. URL: github.com/Cloud-Architekt/AzureAD-Attack-Defense/tree/main (visited on 10/07/2025) (cit. on p. 17).

[45] Nicola Zannone, Emmanuele Zambon-Mazzocato, and Habib Mostafaei. "Security Analysis of Azure Active Directory Logs using Machine Learning". In: (2023) (cit. on pp. 17, 18).

[46] Ömer Aslan and Refik Samet. "A Comprehensive Review on Malware Detection Approaches". In: *IEEE Access* 8 (Jan. 2020), pp. 1–1. DOI: 10.1109/ACCESS.2019.2963724 (cit. on p. 18).

[47] Daniel Arp, Michael Spreitzenbarth, Malte Hübner, Hugo Gascon, and Konrad Rieck. "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket". In: Feb. 2014. DOI: 10.14722/ndss.2014.23247 (cit. on p. 18).

[48] Alessandra Gorla, Ilaria Tavecchia, Florian Gross, and Andreas Zeller. "Checking app behavior against app descriptions". In: *Proceedings of the 36th International Conference on Software Engineering*. ICSE 2014. Hyderabad, India: Association for Computing Machinery, 2014, pp. 1025–1035. ISBN: 9781450327565. DOI: 10.1145/2568225.2568276. URL: https://doi.org/10.1145/2568225.2568276 (cit. on p. 18).

[49] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation". In: *J. Mach. Learn. Res.* 3.null (Mar. 2003), pp. 993–1022. ISSN: 1532-4435 (cit. on p. 18).

[50] Maarten Grootendorst. *BERTopic: Neural topic modeling with a class-based TF-IDF procedure*. 2022. arXiv: 2203.05794 [cs.CL]. URL: https://arxiv.org/abs/2203.05794 (cit. on pp. 18, 19, 38).

[51] Nishavi Ranaweera, Jiarui Xu, Suranga Seneviratne, and Aruna Seneviratne. *BERTDetect: A Neural Topic Modelling Approach for Android Malware Detection*. 2025. arXiv: 2503.18043 [cs.CR]. URL: https://arxiv.org/abs/2503.18043 (cit. on pp. 18–20, 38).

[52] Xiaobao Wu, Thong Nguyen, and Anh Tuan Luu. "A survey on neural topic models: methods, applications, and challenges". In: *Artificial Intelligence Review* 57.2 (Jan. 2024). ISSN: 1573-7462. DOI: 10.1007/s10462-023-10661-7. URL: http://dx.doi.org/10.1007/s10462-023-10661-7 (cit. on p. 19).

[53] Martin Nguyen and Tirath Ramdas. *Topic Modeling: A Comparative Overview of BERTopic, LDA, and Beyond*. July 2025. URL: chamomile.ai/topic-modeling-overview/ (visited on 08/15/2025) (cit. on p. 19).

[54] Marco Alecci, Jordan Samhi, Tegawendé Bissyandé, and Jacques Klein. "Revisiting Android App Categorization". In: Apr. 2024, pp. 1–12. DOI: 10.1145/3597503.3639094 (cit. on p. 19).

[55] Microsoft Learn. *Viewing Apps Using Your Tenant for Identity Management - Microsoft Entra ID*. Oct. 2023. URL: learn.microsoft.com/en-us/entra/identity/enterprise-apps/application-list (visited on 09/23/2025) (cit. on p. 21).

[56] Microsoft Learn. *List TeamsApp - Microsoft Graph V1.0*. July 2025. URL: learn.microsoft.com/en-us/graph/api/appcatalogs-list-teamsapps?view=graph-rest-1.0&tabs=http (visited on 09/23/2025) (cit. on p. 24).

[57] Selenium. *The Selenium Browser Automation Project*. URL: www.selenium.dev/documentation/ (cit. on p. 27).

[58] michevnew. *PowerShell/App_Permissions_inventory_GraphAPI.md at Master · Michevnew/PowerShell*. 2018. URL: https://github.com/michevnew/PowerShell/blob/master/app_Permissions_inventory_GraphAPI.md (visited on 09/24/2025) (cit. on p. 28).

[59] Microsoft Learn. *Get Started with the Microsoft Graph PowerShell SDK*. July 2025. URL: learn.microsoft.com/en-us/powershell/microsoftgraph/get-started?view=graph-powershell-1.0 (visited on 09/24/2025) (cit. on p. 29).

[60] Tenable. *Dangerous Delegated Permissions Affecting the Tenant*. 2025. URL: www.tenable.com/indicators/ioe/entra/DANGEROUS-DELEGATED-PERMISSIONS-AFFECTING-THE-TENANT (visited on 10/07/2025) (cit. on p. 32).

[61] Tenable. *Dangerous Application Permissions Affecting the Tenant*. 2025. URL: www.tenable.com/indicators/ioe/entra/DANGEROUS-APPLICATION-PERMISSIONS-AFFECTING-THE-TENANT (visited on 10/07/2025) (cit. on p. 32).

[62] UMAP Learn. *Basic UMAP Parameters — Umap 0.5 Documentation*. URL: umap-learn.readthedocs.io/en/latest/parameters.html (cit. on p. 36).

[63] Hugging Face. *Sentence-Transformers/All-MiniLM-L6-v2 · Hugging Face*. URL: huggingface.co/sentence-transformers/all-MiniLM-L6-v2 (cit. on p. 36).

[64] scikit-learn developers. *Sklearn.cluster.HDBSCAN*. URL: scikit-learn.org/stable/modules/generated/sklearn.cluster.HDBSCAN.html (cit. on p. 36).

[65] Wikipedia Contributors. *Silhouette (Clustering)*. Aug. 2019. URL: en.wikipedia.org/wiki/Silhouette_(clustering) (visited on 06/10/2024) (cit. on p. 38).

[66] GeeksforGeeks. *Understanding OneClass Support Vector Machines*. Feb. 2024. URL: www.geeksforgeeks.org/machine-learning/understanding-one-class-support-vector-machines/ (visited on 06/10/2024) (cit. on p. 40).

[67] Pramod kumar. *Understanding LOF (Local Outlier Factor) for Implementation*. July 2020. URL: medium.com/@pramodch/understanding-lof-local-outlier-factor-for-implementation-1f6d4ff13ab9 (visited on 06/10/2024) (cit. on p. 41).

[68] Cory Maklin. *Isolation Forest*. July 2022. URL: medium.com/@corymaklin/isolation-forest-799fceacdda4 (cit. on p. 42).

# Appendix A

# Appendix A

**Listing A.1:** Example of JSON representation of an application manifest from `/tiledata/` MS AppSource API

```
    "dataList": [
      {
        "entityId": "WA200005271",
        "entityIdLoweredCased": "wa200005271",
        "entityType": 1,
        "title": "ChatGPT for Excel",
        ...
      },
      {
        "entityId": "WA200000113",
        "entityIdLoweredCased": "wa200000113",
        "entityType": 1,
        "title": "draw.io Diagrams",
        ...
      },
        ...
    ]
```

**Listing A.2:** Example of JSON representation of an application manifest from `/view/app/` MS AppSource API

```
   {
    "entityId": "WA200005271",
    ...
    "title": "ChatGPT for Excel",
    "Description": "The Ultimate ChatGPT for Excel Add-in: ... Automate
    , analyze, and accelerate your spreadsheets, ...",
    "shortDescription": "Best AI Excel Assistant",
    "popularity": 4.481,
    "ratingSummary": {
        "averageRating": 4.481,
        "TotalRating": 511
    },
    "categoriesDetails": [
        {
```

```
            "longTitle": "Analytics",
            ...
        },
        ...
    ],
    "licenseManagement":{
        "isMicrosoftManaged": false,
        ...
    }
    ...
   }
```

**Listing A.3:** Example of JSON representation of a Teams application with its permissions

```
{
    "id": "5db56dd0-534d-467b-aeda-d622bee2574a",
    ...
    "appDefinitions": [
        {
            "id": "<GLOBAL_ID>",
            ...
            "displayName": "Q&A",
            ...
            "shortDescription": "This app...",
            "description": "The Q&A app is a solution for...",
            ...
            "authorization": {
                "clientAppId": "00000005-0000-0ff1-ce00-000000000000",
                "requiredPermissionSet": {
                    "resourceSpecificPermissions": [
                        {
                        "permissionValue": "OnlineMeeting.ReadBasic.
    Chat",
                        "permissionType": "delegated"
                        },
                        {
                        "permissionValue": "ChannelMeeting.ReadBasic.
    Group",
                        "permissionType": "delegated"
                        }
                    ]
                }
            }
        }
    ]
}
```

**Listing A.4:** Example of JSON representation of a SharePoint application with its permissions

```
{
    ...
```

```
    "WebApiPermissions": "Microsoft Graph, User.Read; Microsoft Graph,
    User.Read.All; Microsoft Graph, User.ReadBasic.All",
    ...
}
```

**Listing A.5:** Python code for heuristic risk scoring of application permissions

```
PERMISSION_WATCHLIST = {
    "AdministrativeUnit.ReadWrite.All": 25,
    "Application.ReadWrite.All": 25,
    "AppRoleAssignment.ReadWrite.All": 25,
    "Directory.ReadWrite.All": 25,
    "Group.ReadWrite.All": 25,
    "User.ReadWrite.All": 15,
    "User.DeleteRestore.All": 25,
    "RoleManagement.ReadWrite.Directory": 25,
    "Policy.ReadWrite.All": 25,
    "Sites.FullControl.All": 20,
    "Mail.ReadWrite.All": 15,
    "Files.ReadWrite.All": 15,
    "Directory.Read.All": 10,
    "User.Read.All": 10,
    "Mail.Read.All": 10,
    "Files.Read.All": 10,
    "AuditLog.Read.All": 15,
    "ChannelMessage.Read.All": 15,
    "Chat.Read.All": 15,
    "Calendars.Read.All": 10,
    "Contacts.Read.All": 10,
    "EWS.AccessAsUser.All": 10,
}

# general rules and patterns
RULE_WEIGHTS = {
    'base_attack_surface': 0.5, # Score per permission
    'is_application_perm': 5,   # Added if permission type is '
    Application'
    'is_rsc_perm': -5,          # Benefit for using Resource-Specific
    Consent
    'has_readwrite_pattern': 3, # General risk for ReadWrite
    permissions
    'has_all_pattern': 5,       # General risk for '.All' permissions
}


def calculate_risk_score(permissions_list):
    """
     Calculates a heuristic risk score for an app based on its
    permissions.
    """
    score = 0
    if not isinstance(permissions_list, list):
        return 0
```

```
43
44      # base score for the total number of permissions
45      base_score = len(permissions_list) * RULE_WEIGHTS['
        base_attack_surface']
46      score += base_score
47
48      # Add score based on specific watchlist permissions and general
        patterns
49      for perm_string in permissions_list:
50
51          # string parsing
52          parts = perm_string.split('-')
53          perm_key = parts[0]
54          perm_type = parts[-1] if len(parts) > 2 else 'Unknown'
55
56
57          if perm_key in PERMISSION_WATCHLIST:
58              score += PERMISSION_WATCHLIST[perm_key]
59
60          # Add score based on general rules and patterns
61          if perm_type.lower() == 'application':
62              score += RULE_WEIGHTS['is_application_perm']
63
64          if "Resource-Specific Consent (RSC)" in perm_string:
65              score += RULE_WEIGHTS['is_rsc_perm']
66
67          # Add score for general risky patterns if not already in the
        main watchlist
68          if perm_key not in PERMISSION_WATCHLIST:
69              if 'readwrite' in perm_key.lower():
70                  score += RULE_WEIGHTS['has_readwrite_pattern']
71              if '.all' in perm_key.lower():
72                  score += RULE_WEIGHTS['has_all_pattern']
73
74      # Ensure score does not fall below zero
75      final_score = max(base_score, score)
76
77      return round(final_score, 2)
```

# Appendix B

# Appendix B

| ID | Top-5 Terms | Interpretation | Count |
|---|---|---|---|
| 0 | (crm, file, template, attachment, send) | File Management | 122 |
| 1 | (360, student, learning, course, training) | E-Learning | 65 |
| 2 | (ticket, notification, onboarde, core, incident) | Notifications | 65 |
| 3 | (office, room, book, desk, space) | Workspace Booking | 59 |
| 4 | (asset, security, permission, software, expense) | Security | 75 |
| 5 | (calendar, event, meeting, appointment, scheduling) | Scheduling & Calendar | 63 |
| 6 | (idea, board, meeting, visual, brainstorm) | Collaborative Brainstorming | 62 |
| 7 | (sharepoint, page, card, viva, connection) | SharePoint Integration | 49 |
| 8 | (birthday, web, sharepoint, webpart, navigation) | Intranet & WebParts | 45 |
| 9 | (signature, sign, document, signing, digital) | Digital Signatures | 33 |
| 10 | (task, project, checklist, board, gantt) | Project Management | 50 |
| 11 | (sms, call, communication, message, agent) | Communication Tools | 31 |
| 12 | (recognition, culture, employee, recognize, peer) | Employee Recognition | 29 |
| 13 | (ai, knowledge, answer, enterprise, unleash) | AI Knowledge Base | 50 |
| 14 | (meeting, ai, summary, transcript, meet) | AI Meeting Assistants | 37 |
| 15 | (survey, video, feedback, quiz, meeting) | Surveys & Feedback | 28 |
| 16 | (break, copilot, habit, journey, movement) | Well-being & Productivity | 36 |
| 17 | (language, translation, meeting, multilingual, ai) | AI Translation | 19 |
| 18 | (confluence, tab, view, analytic, meeting) | Confluence Integration | 55 |
| 19 | (form, approval, travel, sharepoint, component) | Forms & Workflows Design | 25 |
| 20 | (pdf, pdfs, document, convert, file) | PDF Management | 15 |
| 21 | (directory, employee, people, hr, profile) | Human Resources | 20 |
| 22 | (expense, project, task, budget, claim) | Expense Management | 24 |
| 23 | (legal, ai, firm, matter, capture) | Legal AI Assistant | 12 |

**Table B.1:** Topic Representative Words, Interpretation, and Applications Count

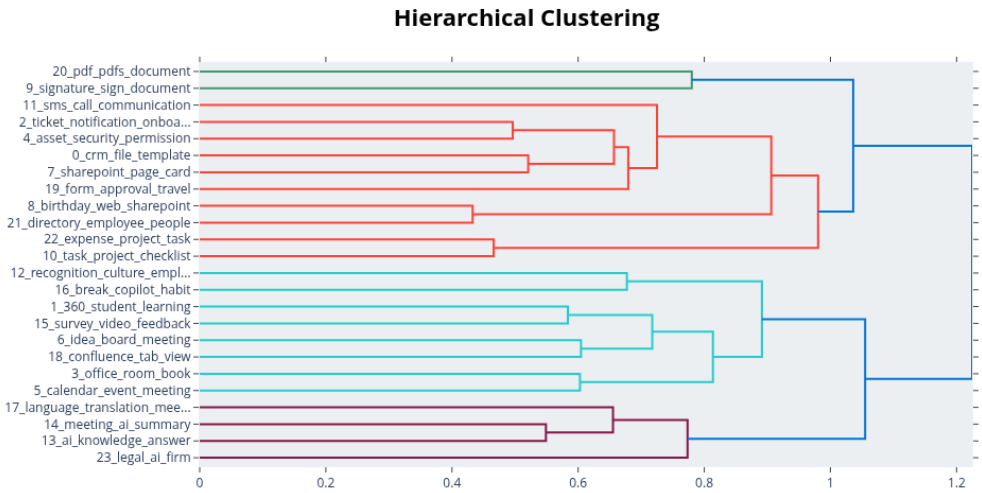


**Figure B.1:** Hierarchical Clustering based on Topic Embeddings

| Algorithm | Strategy | Best parameter | Avg AUC |
|---|---|---|---|
| OC-SVM | Cross Easy | `nu`=0.2 | 0.618 |
| OC-SVM | Cross Medium | `nu`=0.01 | 0.535 |
| OC-SVM | Cross Hard | `nu`=0.2 | 0.488 |
| OC-SVM | Synthetic Easy | `nu`=0.01 | 0.864 |
| OC-SVM | **Synthetic Medium** | `nu`=0.1 | **0.760** |
| OC-SVM | Synthetic Hard | `nu`=0.05 | 0.634 |
| LOF | Cross Easy | `k`=3 | 0.436 |
| LOF | Cross Medium | `k`=3 | 0.482 |
| LOF | Cross Hard | `k`=3 | 0.500 |
| LOF | **Synthetic Easy** | `k`=3 | **0.694** |
| LOF | Synthetic Medium | `k`=3 | 0.632 |
| LOF | Synthetic Hard | `k`=3 | 0.517 |
| IForest | Cross Easy | `n_estimators`=50 | 0.419 |
| IForest | Cross Medium | `n_estimators`=100 | 0.470 |
| IForest | Cross Hard | `n_estimators`=150 | 0.498 |
| IForest | **Synthetic Easy** | `n_estimators`=150 | **0.785** |
| IForest | Synthetic Medium | `n_estimators`=100 | 0.713 |
| IForest | Synthetic Hard | `n_estimators`=150 | 0.642 |

**Table B.2:** Best hyperparameters and corresponding AUC for each algorithm and strategy. Bold indicates the overall best configuration for each algorithm.