



Classifying Financial Transaction for Invoice Matching

Massimiliano Zanoni
Evgenii Veksin

OCTOBER 2026

This is a creation of **iason**.

The ideas and model frameworks described in this document are the result of the intellectual efforts and expertise of the people working at **iason**. It is forbidden to reproduce or transmit any part of this document in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of a company in the **iason Group**.



Research Paper Series

Year 2026 - Issue Number 90

Last published issues are available online:

<http://www.iasonltd.com/research>

Front Cover: **Umberto Boccioni**, *La città che sale*, 1910.



ESSENTIAL SERVICES FOR
FINANCIAL INSTITUTIONS



Executive Summary

This work addresses the automated classification of bank transactions to facilitate the reconciliation of invoices, a critical and often tedious task in the financial control process. The study evaluates two primary methodologies: a Machine Learning approach known as K-Nearest Neighbors (k-NN), based on the Levenshtein distance, and a modern Large Language Model (LLM) approach based on the BERT architecture. To manage computational efficiency and scalability of the Machine Learning approach, the work explores Hierarchical Agglomerative Clustering (HAC) and OPTICS, which successfully reduced the training dataset size by 5 to 6 times while maintaining an accuracy level of approximately 99%. The LLM component utilizes a hybrid framework combining BERT embeddings with Bag-of-Words (BoW) representations to effectively handle low-data environments and the specific nuances of financial jargon and abbreviations. Experimental results for both methodologies demonstrate accuracy exceeding 99%, rivaling traditional manual rule-based approaches.

Furthermore, the study introduces confidence metrics, derived from average neighbor distances in the k-NN model or probability distributions in the LLM approach, to selectively spot "doubtful" cases for manual review. In conclusion, the integration of these AI and ML techniques provides a precise, consistent, and scalable solution, allowing accountants to focus on higher-value analytical activities.

About the Authors

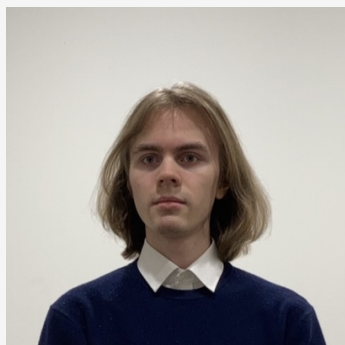


Massimiliano Zanoni:

Senior Manager

With over 20 years experience in the management of complex projects in credit and financial risk he has worked within leading national and international financial services, maturing a combination of quantitative analysis and organizational skills. His consulting experience ranges from model development to the implementation of management reporting system; from to compliance and validation of IRB credit risk framework to setup of ALM systems. He has worked for Veneto Banca as the Head of Methodology and Risk Management Reporting. In the same period, he was also the delegate in the Risk committees of foreign banks and the Risk Management delegate in the Board of Banca Apulia where he directly oversaw the transition process to the Solvency II framework of the participated insurance company. In iason, he has been involved in the development of credit risk solutions and the implementation of a Dynamic Balance Sheet framework.



**Evgenii Veksin:***Analyst*

He holds bachelor's degree in economics at Higher School of Economics (Moscow). Secondly, he holds a master's degree in Management Engineering at Politecnico di Milano. In Iason, he was involved on the Credit Risk topics: Stress Test project, internal GRisk models.



This article was written in collaboration with Massimiliano Zanoni, who at the time were working for Iason Consulting.

Table of Content

Introduction	p.6
Dataset and Rule-based Classification	p.7
The K-Nearest Neighbors Approach	p.8
Introducing the Levenshtein Distance with k-NN	p.9
Optimizing the Training Dataset	p.11
Test Results and Conclusions	p.15
Classifying Transaction with LLM	p.18
Tokenization	p.19
The SpaCy Environment	p.20
The Bag-of-Words Representation and the BERT Embedding	p.21
Test Results and Conclusions	p.24
References	p.26

THE process of reconciling invoices with corresponding bank transactions serves as a fundamental control mechanism to ensure consistency between a company's internal financial records and the official statements issued by its banking institutions. This reconciliation is essential to guarantee that accounts payable and accounts receivable accurately represent the underlying cash movements recorded by the bank. This can be challenging and time-consuming and has introduced areas for automated support across invoice processing tasks and research provides evidence that the use of hierarchical information can highly improve classification precision [9].

The effectiveness of this procedure depends critically on the correct classification of bank transactions, as misclassifications can compromise the automation of the matching process and lead to discrepancies, manual interventions, and potential accounting errors. More specifically, the matching activity involves the systematic analysis of a large amount of structured and unstructured data, in particular:

- **Invoices recorded in the company's accounting system**, representing the formal documentation of outstanding obligations to suppliers (accounts payable) and amounts due from customers (accounts receivable). These invoices reflect the organization's official financial commitments and claims¹.
- **Bank statement lines**, i.e. the official bank transaction descriptions recorded by the bank, which can be collected from an external servicer, upon authorization.

The precise synchronization between expected and actual payment datasets facilitates the automated identification of financial correspondences. Within this framework, the granularity and accuracy of bank transaction classification emerge as critical prerequisites for improving the reliability, efficiency, and scalability of automated invoice matching.

High-performance classification tools are essential for isolating invoice-related entries from extraneous operational "noise"-such as commissions, taxes, or interest charges. By effectively filtering these datasets, such tools significantly constrain the search space for potential matches, thereby streamlining the reconciliation process and reducing the risk of false-positive correspondences. With increasing data volumes and the demand for rapid and accurate decision-making, the integration of AI and ML has become pivotal. [2] Specifically, the application of machine learning has the potential to provide novel insights into empirical data and to improve predictive performance. [7]

This study focuses on the use of AI in the classification of bank transactions. Through the application of Machine Learning (ML) methods and Large Language Model (LLM) techniques, it explores the benefits and challenges of adopting these technologies, focusing on their potential to enhance efficiency and accuracy within management accounting as in [2], and shows that AI-based systems supplement human skills, allowing accountants to focus on higher-value jobs as in [13]. The study is organized as follows:

- **Chapter 1** examines a K-nearest neighbors (KNN) model based on the Levenshtein distance, to extract meaningful information from transaction descriptions and classify the transactions under analysis. The chapter further evaluates two strategies for reducing and optimizing the training set, by statistically aggregating similar descriptions into homogeneous clusters from which representative points are selected to construct a reduced training set. The performance of the KNN classifier is then assessed across multiple reduced training configurations, to determine the extent to which the the

¹In the Italian context, where invoices are digitally recorded by the Internal Revenue Service, these can be downloaded from "Cassetto Fiscale", after the firm has authorized the servicer.

Keywords in Description	Transaction Class	Transaction Type
"ACCR.IMP."	Pending Invoice	Total
"ANT FATT O DOC ITA"	Pending Invoice	Pending
"ACCR.BEU", "BONIFICO A VOSTRO FAVORE", "BONIFICO A VS.FAV."	Transfer	Credit
"BEU", "BON.UE", "BONIF."	Transfer	Charge
"PAGAMENTO ADUE"	Transfer	Payment Order
"DISP.BEU STIP."	Transfer	Salary
"STORNO"	Transfer	Reversal Movement
.....		

TABLE 1: *Keyword Based Association.*

process can serve as a viable approach when the size of the transactions set grows significantly.

- **Chapter 2** extends the analysis by introducing modern LLM-based techniques, specifically those derived from the BERT architecture, to enhance traditional transaction classification methods. In contrast to classical models relying on surface-level text matching, advanced natural language processing (NLP) algorithms can capture the contextual meaning of textual components, enabling more nuanced interpretations. This capability is particularly relevant for financial transactions, where the sequence of keywords may play a decisive role in determining the correct classification.

Dataset and Rule-based Classification

The dataset includes more than 800 bank transactions relative to one quarter activity from the bank accounts of the selected firm. Such a dataset clearly is not suitable for big data analysis, but it may be adequate to train a ML model in order to recognize existing text structures and possibly interpret new descriptions.

The description of account transactions is clearly bank specific and similar transactions may present different descriptions, i.e. the text may or may-not include the counterpart name, the invoice number, the reference date or other technical codes. In addition, the structure of descriptions is highly dependent on the category of the transaction. Descriptions of technical transactions, like commissions or interests, are typically rather homogeneous in comparison to business transactions, for example the following descriptions represent the same type of business transaction:

- ßEU INTERN BANK Bonifico da Voi disposto a favore di;
- ßON.UE CAN.TELEM. Bonifico da Voi disposto a favore di:

At first, transactions were classified through manually defined rules, based on defined keywords: each time a set of keywords was found in the description field, the transaction was classified accordingly. The following Table 1 lists a sample of keywords and their relative rule-based classification.

This rule-based approach is simple to understand and works reasonably well, but it lacks flexibility and requires a considerable effort. Each new bank is associated with a specific set

of manually defined keywords and rules whose assignment is considered as the "correct" reference. Creating and maintaining these training sets involves a significant manual effort, tailored to each new bank, and requires updates whenever a bank changes the format or style of its transaction descriptions. To automatize the identification of keywords and the classification of transactions, a Machine Learning approach, based on K-Nearest Neighbors and Levenshtein distance, was tested with surprising results. The following paragraph describes the methodology adopted.

The K-Nearest Neighbors Approach

This chapter introduces the k-Nearest Neighbors algorithm, a fundamental technique in machine learning that is widely applied to classification problems, including those involving textual data. The central idea behind k-NN is straightforward: when a new input sample is presented, the algorithm identifies the k training instances that are most similar to it, according to a chosen distance metric. The class of the new sample is then inferred from the majority class among these nearest neighbors.

In text classification tasks, the notion of "distance" is typically interpreted as a measure of linguistic similarity. One common approach is to use the **Levenshtein distance**, which calculates the minimum number of character-level edits or insertions, deletions or substitutions, required to transform one string into another. This metric effectively captures how similar two short text strings are, making it suitable for tasks such as matching textual labels or categorizing brief messages.

Formally, k -NN is a **non-parametric, instance-based supervised learning algorithm**. Unlike parametric models, it does not build an explicit predictive model during training. Instead, it simply stores the training data and defers all computations to the prediction phase. When classifying a new instance, the algorithm relies directly on the stored data to determine the appropriate class label.

Due to its simplicity, transparency, and ease of implementation, k-NN is frequently used as a **baseline method** in natural language processing applications, particularly those involving the classification or clustering of short text segments.

The k-Nearest Neighbors algorithm works by finding the closest number (k) of training samples to the new sentence which needs to be classified, in terms of the defined distance. As any supervised it can be split into two phases:

- **Training Phase:** in contrast to model based algorithms, k-NN does not involve parameter estimation nor gradient-based optimization. The training phase consists solely of storing the entire dataset of labeled transaction descriptions. Each description retains its assigned class and serves as a reference point for subsequent comparisons.
- **Prediction Phase:** when a new, unclassified transaction description is provided, the algorithm computes its distance to every stored training instance. It then identifies the k closest descriptions, those exhibiting the greatest textual similarity according to the selected metric. The class most frequently represented among these k neighbors is subsequently assigned to the new description. This majority-voting mechanism enables k-NN to leverage local patterns in the data, making it particularly effective when similar descriptions consistently correspond to the same transaction category.

By relying directly on the structure of the training samples and their linguistic proximity, the k-NN algorithm provides an interpretable and flexible baseline method for transaction

classification. Its ability to generalize from local patterns makes it especially well-suited for environments where descriptions are heterogeneous yet recurrent, such as bank statements. Through these two phases, the k-NN algorithm provides a straightforward yet powerful framework for transactions classification, laying the groundwork for more advanced techniques explored in the following sections. This chapter is organized as follows:

- A. Introducing the Levenshtein Distance with k-NN;
- B. Optimizing the Training Dataset;
- C. Presentation of Results;
- D. Conclusions.

Introducing the Levenshtein Distance with k-NN

The key to applying k-NN to text classification, is to define a robust measure of the difference between two sentences. In this work the Levenshtein distance was adopted for this purpose. It provides the difference between two sentences in terms of the number of single-character edits (insertions, deletions, or substitutions) required to change one string into the other. When using this methodology, a strong assumption is made, i.e. descriptions that are close are also semantically similar, hence they can be classified with the same tag. A refinement of the standard Levenshtein distance is its normalized form, calculated as the standard distance (number of edits) divided by the length of the longer string. By definition, this measure ranges between 0 and 1, i.e. it is 0 if the two strings are identical and 1 if every character has to be changed to obtain the test description. The classification process is described by the following steps:

- 1 Select the training Dataset, i.e. the list of transaction description classified with a deterministic rule-based approach (example in Table 1).
- 2 Chose the test sentence, i.e. a new transaction description to be classified.
- 3 Calculate the distances between the new description and every single sentence in the training dataset. One of the following different cases may occur:
 - a) The two sentences belong to the same category, "accr.imp. non ant. cod. disposiz: saldo fattura n. 11". " accr.imp. non ant. cod. disposiz: saldo fattura n. 22". Levenshtein dist. = 2; Normalized dist. = 0.037.
 - b) The two sentences partially overlap but belong to different categories, "accr.imp. non ant. cod. disposiz: saldo fattura n. 11", " accr. beu cod. disp.: cash bon.eu" the Levenshtein dist. = 31 , Normalized dist. = 0.58.
 - c) The two sentences have no similarity thus belong to different categories, "accr.imp. non ant. cod. disposiz: saldo fattura n. 11", "pagamento rata mutuo/prestito pagamento rata n. 9 del finanzia") the Levenshtein dist. = 51 , Normalized dist. = 0.83.
- 4 Find the closest neighbors, i.e. the closest k sentences in the training set, those having the lowest Levenshtein distance to the test sentence.
- 5 Assign the class; the most frequent tag found among the k-nearest neighbors is selected as the correct label to be assigned to the test sentence. In the case of K=3, the tag selected is that of the majority among neighbors.

Advantages	Disadvantages
1. <u>Minimal training</u>: the "training" simply consists in storing the data, making it very fast to "train" the model. The computational work is all done at prediction time (i.e. measuring distances). 2. <u>Simple and Intuitive</u>: the approach represents a natural way to think about similarity, the more alike two strings are, the closer their distance. 3. <u>Good for small datasets</u>: this method can perform well with small datasets where the relationships between sentences are not too complex.	1. <u>Computational Cost</u>: calculating the Levenshtein distance between a new sentence and every single sentence in a large dataset is computationally burdening. This is a major drawback for real-time applications with a high volume of data. 2. <u>Scalability Issues</u>: as the training dataset grows, the prediction time increases linearly, making it a poor choice for large-scale production systems. 3. <u>Overemphasis on Character Order</u>: Sentences that are semantically very similar might have a high Levenshtein distance due to different word choices or sentence structures. For example, "How do I turn on the light?" and "Can you switch the light on?" have a high Levenshtein distance but a similar meaning.

TABLE 2: *k*-NN Advantages and Disadvantages.

Advantages and disadvantages of the *k*-NN approach are summarized in Table 2. The first two disadvantages are successfully addressed in this work, as described in the following paragraphs. The third disadvantage is more relevant in the context considered, however, it should be efficiently addressed with the the LLM approach analyzed in the next chapter.

The training dataset may include transactions with rare classifications or even occurring only once (e.g , "bonifico storno", see Table 4). Clearly, such cases cannot be classified correctly with the current approach since there are not enough neighbors with the same category to be selected and determine the correct classification. Therefore all transactions with classifications occurring only once or twice in the dataset will be excluded from the test, but still considered in the training set.

The Levenshtein distance of descriptions close to the transaction in analysis is improved by eliminating from the text irrelevant components, which generate noise, as described by the following procedure:

- Convert every text to lower case, as the case type does not change the meaning of keywords.
- Delete large meaningless sequences, not useful for classification purposes but adding noise to the analysis, such as strings containing at least 6 digits. Example is presented in line **(1)** of Table 3.
- Truncate descriptions, usually key information is located in the first half of the text, whereas the second half includes transaction specific data, such as counterparty name,

	Before	After
(1)	bon.ue can.telem. 0124092836794514 w0242102615507532400000272	bon.ue can.telem.
(2)	bonifico a vostro favore bonifico sepa da banca popolare di puglia e Basilicata società cooperativa per azioni per a saldo vostra fattura 144 20/12/202 trn	bonifico a vostro favore bonifico sepa da banca popolare di puglia e Basilica
(3)	ACCR. BEU COD. DISP.: 0124092731001227 CASH BON.EUR.UNICO VS.FAV. CON CONTAB. NOTPROVIDED 78 20240730	accr. beu cod. disp.: cash bon.eur.un

TABLE 3: *Examples of Description Cleaning.*

invoice number, etc. not relevant for classification. This approach reduces the noise but is mainly valid for long description of the type presented in the line (2) of table 3.

The three lines in Table 3 provide an example of the effect of applying the previous rules.

Procedure Description

To maximise the size of the training set used in this exercise, a leave-one-out validation approach was adopted. Under this procedure, each transaction is tested individually using all remaining transactions as the training set. Concretely, for each of the 836 original descriptions, one description is removed and treated as the test instance. The Levenshtein distance is then computed between this test description and the remaining 835 transactions to identify the three closest matches. The test transaction is subsequently assigned to the most frequently occurring class among these three (i.e., the class appearing in at least two out of the three cases).

If all three neighbours belong to different classes, the classification is determined by the single transaction with the smallest distance to the test description. As noted previously, transactions associated with labels that appear only once or twice cannot be included in the test set, since they cannot be meaningfully evaluated within the leave-one-out procedure. In this dataset, only two such transactions exist (see Table 4), each appearing only once. Consequently, 834 transactions remain eligible for testing under the leave-one-out framework. Comparing each test description against all others becomes computationally burdensome in large datasets, and the efficiency of the process decreases as the training set grows. Given that many descriptions are highly similar, computing the distance to each individual instance is suboptimal. A potential improvement would involve reducing the effective training set by clustering similar descriptions and subsequently computing distances only against representative elements of each cluster rather than against all individual transactions. This shifts the challenge from calculating a full pairwise distance matrix to developing a robust and statistically sound method for aggregating similar transactions into coherent groups, ideally containing transactions belonging to the same classification. The following paragraphs describe and compare two different grouping methodologies.

Optimizing the Training Dataset

Many transaction descriptions differ only for minor details which, for the purpose of classification, can be neglected. This means that transactions could be grouped into homogeneous bundles, to reduce computation effort and time, maintaining the information necessary to correctly classify them.

Full Description	Truncated Description	Classification
"PAGAM. TRAMITE POS COOP.RADIO TAXI PADOVA VIA 13/05-14:29 - Carta n.5259XXXX XXXX XX07ABI: 09517 COD.3058793/000165"	"pagam. tramite pos coop.radio taxi padova via13/0"	None
"STORNO BEU STORNO DEL BONIFICO RIF.71 – 0125011037593015 CONTENUTO NEL FLUSSO RIF.DICEMBRE 2024"	"storno beu storno delbonifico rif.71 -"	Bonifico storno

TABLE 4: *Rarely Occuring Transactions.*

The structure of the dataset is more effectively visualized using the Multidimensional Scaling (MDS) algorithm, which provides a geometric representation of the data by taking as input the matrix of pairwise Levenshtein distances and projecting the observations into an abstract Cartesian space (see Figure 1). The objective of MDS is to preserve the original similarities between data points: transactions that are highly similar in their textual description (i.e., with small Levenshtein distances) are mapped to nearby locations, whereas dissimilar transactions are placed farther apart. Through this transformation, recurring structures in data, such as clusters, gradients, and anomalies, become visually apparent, even though they may be difficult to detect when examining the raw textual data directly. This representation allows:

- To highlight the relative ‘density’ of groups, showing which classes of transactions are numerous;
- To mark isolated (i.e. exceptional) transactions which may signal anomalies or simply outliers.

To create this homogeneous samples, two different clustering approaches are tested, both based on the Levenshtein Distance Matrix, listed together with the sampling procedure:

- a. Hierarchical Agglomerative;
- b. OPTICS clustering;
- c. Sampling Procedure.

a. Hierarchical Agglomerative Clustering

Hierarchical Agglomerative Clustering (HAC) is a bottom-up clustering methodology that allows to construct a tree-like hierarchical structure by iteratively merging data points or clusters based on their pairwise similarity. The process begins by treating each observation as an individual cluster. At each successive step, the two clusters exhibiting the smallest dissimilarity, typically defined by a chosen distance metric, in this study the Levenshtein distance, are merged to form a new, larger cluster. This iterative aggregation continues until all observations have been combined into a single cluster, thereby producing a complete hierarchical representation of the dataset. This approach is particularly effective for text-based similarity analyses because it makes the underlying structure of the data explicit.

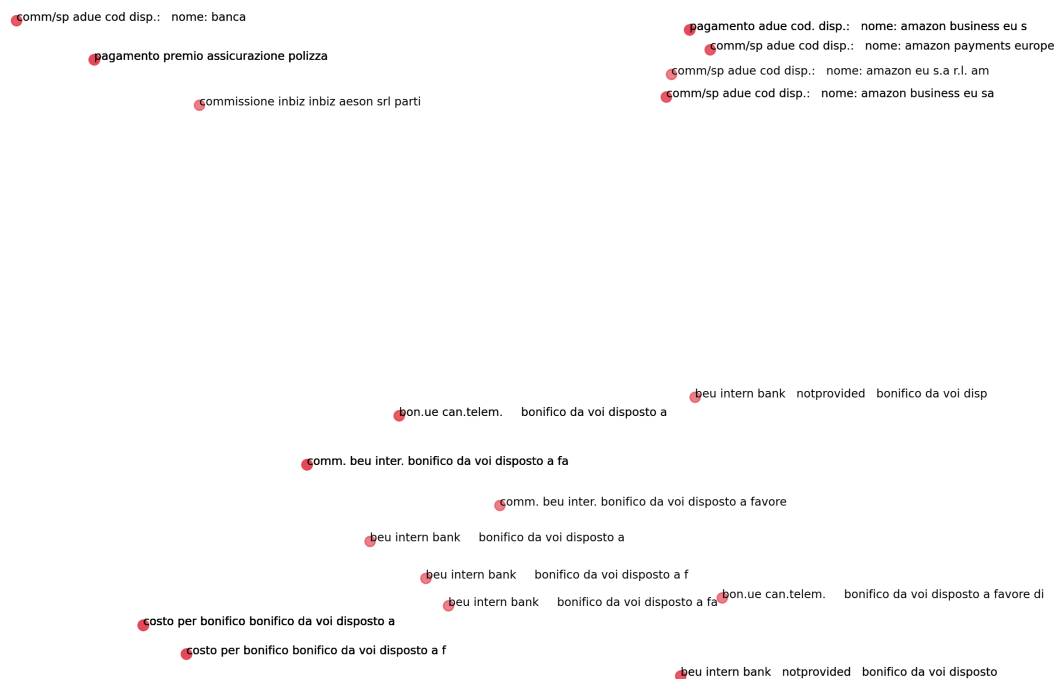


FIGURE 1: MDS Visualization of a Portion of the Transactions Dataset.

HAC captures similarity relationships at multiple levels of granularity: initially, highly similar descriptions are grouped into small, tight clusters; subsequently, these clusters are merged into progressively broader groupings. The resulting hierarchy is visualized through a **dendrogram** (see Figure 2), which provides an interpretable graphical summary of the clustering process, with the horizontal axis representing the distance at which clusters are merged, while individual descriptions are arranged along the vertical axis. Branching points indicate cluster separations, and the height of each branch reflects the dissimilarity between the merged clusters. A key advantage of HAC is that it allows the analyst to determine the desired cluster granularity. Selecting a particular height at which to "cut" the dendrogram, is equivalent to setting a distance threshold, hence the clustering process can be stopped at the desired level. A lower threshold yields a larger number of small, highly homogeneous clusters, whereas a higher threshold produces fewer, broader clusters. Selecting the optimal threshold is inherently subjective: the literature recognizes that no universal rule exists for determining an optimal cut-off, and that the choice depends heavily on the domain, the distance metric, and the overall analytical objective (Sebastiani P. and Perls, T., 2016 [12]).

In this study, the threshold was selected to create the smallest number of clusters that still contained transactions with coherent and meaningful classifications. The thresholds chosen were 4 for the raw Levenshtein distance and 10% (i.e., 0.1) for the normalized version; transactions with a distance below these respective thresholds were grouped together, reflecting the assumption that such descriptions are sufficiently similar to be considered part of the same cluster.

Overall, HAC provides a flexible and interpretable framework to examine structural patterns in text-based datasets, enabling both fine-grained and high-level insights into similarity relationships among transaction descriptions.

b. OPTICS Clustering

The second grouping approach considered in this study, is the OPTICS algorithm (Ordering

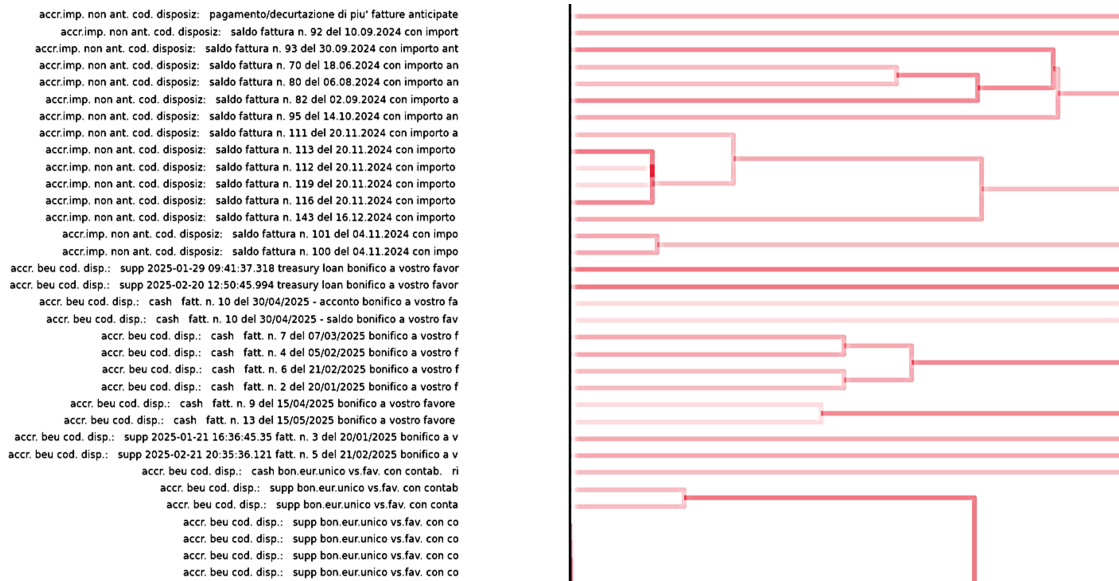


FIGURE 2: Dendrogram Visualization of a Portion of the Transactions Dataset.

Points To Identify the Clustering Structure). This is a primary density-based methodology for database mining, which groups the points in "regions" with high density and defines isolated points as "noise". It does not produce a clustering of a data set explicitly; but instead it creates an augmented ordering of the database representing its density-based clustering structure. [3] It was specifically designed to identify clusters in datasets characterized by heterogeneous density distributions. This capability makes it particularly well suited to the present application, where groups of string descriptions may vary substantially in length and complexity. Unlike algorithms that assume a uniform density across clusters, OPTICS constructs a plot that captures the intrinsic clustering structure at multiple density levels. As a result, it can effectively detect clusters composed of either short or long strings, accommodating the natural variability in string-based distance measures.

A notable advantage of OPTICS is its parsimonious parameterization. The primary parameter to be specified is the minimum number of points required to form a cluster, which controls the granularity and stability of the resulting structure. Conceptually, this parameter plays a role analogous to the distance threshold in hierarchical clustering, as both influence the level at which clusters emerge. In the context of this exercise, setting this parameter is straightforward, it is defined to match the value of K , (i.e. 3) to ensure that clusters consist of at least three related descriptions while maintaining sensitivity to meaningful local density variations.

c. Sampling Procedure

The distribution of data points across groups is markedly uneven. A substantial proportion of transactions is concentrated within a limited number of large groups. Several of these groups contain more than one hundred transactions and are characterized by highly recurrent and standardized descriptions – such as "commissions" or "fiscal transfers" – which appear frequently throughout the dataset. Beyond these dominant groups, the dataset comprises numerous smaller clusters of varying sizes. Finally, a residual set of points remains that cannot be meaningfully assigned to any cluster. These observations are treated as *outliers* due to their idiosyncratic characteristics and low frequency.

Once transactions have been clustered based on their pairwise distances, three points per group are drawn from each group to construct the reduced training set, an arbitrary

choice aligned with the parameter k . To avoid introducing additional noise into the subsequent classification process, it is essential that the selected points faithfully represent the predominant transactional behaviour within each group. For this reason, the sampled points are drawn exclusively from the *most frequent* transaction class within each cluster. The **treatment of outliers** within this sampling procedure must be handled with particular care. While selecting representatives from groups of normal size is straightforward, the handling of points that cannot be aggregated into meaningful clusters differs substantially between clustering methods:

- With hierarchical clustering outliers tend to be aggregated into very small, isolated clusters, ultimately yielding "micro-clusters" of one or very few points. This occurs because the chosen distance threshold prevents them from merging with any larger group.
- On the other hand, OPTICS directly classifies such observations as *noise*, since these points fail to form a valid cluster with the required minimum size of three elements. In this case, therefore, outliers appear explicitly as points that do not satisfy the density conditions necessary for cluster membership.

The question of whether to include these outliers in the representative training sample is non-trivial. On the one hand, outliers generally correspond to rare transaction types, meaning that their inclusion may enhance the model's ability to classify infrequent but potentially important cases. On the other hand, their heterogeneity and scarcity may introduce noise or reduce the quality of the training set. To assess this trade-off empirically, two alternative training samples were constructed and evaluated: one including the outliers identified by the clustering process, and one without them. This dual approach enables a clearer understanding of the impact that rare and irregular transactions have on the overall model performance. The results of the test are presented in the next section.

Test Results and Conclusions

This paragraph presents the results of the exercise highlighting the accuracy of the different sampling approaches, compared to the initial rule-based classification, considered as the correct classification.

The three methods are summarized in the following paragraphs:

- A. The **direct K-NN algorithm**, the full dataset is used as the reference set, i.e. each transaction is compared with all remaining points;
- B. The **Hierarchical algorithm**, transactions are sampled, each test point is compared only with three points from each sample;
- C. The **OPTICS algorithm**, transactions are sampled differently but points are selected as in option B.

Within each sample approach classification is based on the k-NN algorithm using the Levenshtein distance, but the leave-one-out is applied only to option A as, in the others, grouping and the relative sampling is performed once before testing. In case B and C, when the point to be tested belongs to those selected from the clusters, i.e. it is included in the reduced training set, it will be excluded from the training set for this particular case. This process can be easily performed with k-NN, where no pre-determined model is defined.

	Simple k-NN	Hierarchical Agglomerative	OPTICS
	Leave-one-out	Clustering	Clustering
Training set	834(833)	189(188)	177(176)
Outliers #	-	69	28
		Performance with outliers	
Mistakes	6/834	5/834	8/834
Accuracy	99.3%	99.5%	99%
Errors below threshold	-	0/5	2/8
		Performance without outliers	
Training set	Outliers not defined for this method	120(119)	149(148)
Mistakes		9/834	6/834
Accuracy		99%	99.3%
Errors below threshold		0/9	2/8

TABLE 5: Summary of Test Results.

As summarized in the Table 5, with Hierarchical Clustering the size of the final sample reduces to 189 from the original 834 (a 77.4% reduction), while with OPTICS it reduces to 177 including outliers (a 78.7% reduction).

If outliers are excluded instead, the sample reduces to 120 points with Hierarchical Clustering (a 85.5% reduction) and to 149 with OPTICS (a 82.1% reduction) but, in both approaches, the samples size also depends on the choice of parameters. These four samples, with and without outliers, are used as training sets for the classification exercise with the usual k-NN methodology.

As shown in Table 5, the performance are practically the same for both methods with or without outliers, and mirror the results obtained from the Simple k-NN exercise. This proves, beyond any doubt, that the grouping procedure maintains the same good classification performances while significantly reducing the size of the training set.

A detailed analysis of errors can provide information on the boundary between "confident" and "doubtful" classifications and define a rule-of-thumb to activate a direct manual control of doubtful classifications.

Classification Errors

Looking at the classification errors in tables 6 and 7, for each of them we can notice that the average distance of the 3 nearest points is quite higher than for descriptions correctly classified.

Correct classifications present an average distance to their 3 nearest neighbors of about 4.5% for OPTICS, which decreases to about 2.5% for the simple k-NN, while, in case of errors this average increases significantly, often over 50%, apart for 2 cases, (ID 202 and 472). The

Test ID	Test Description	Real Label	Assigned Label	Top 3 ID	Average Distance
84	bonifico a vs.fav. saldo ft 7 del 04.04.25 cro: bonifico n. disposto	transfer, credit	transfer, charge	[2, 89, 204]	61.2%
202	liq. inter. ant.f competen	interests/charges	commissions	[603, 12, 70]	46.4%
472	comm/sp adue cod. nome:banca	commissions	transfer, payment order	[727, 471, 598]	25.7%
601	comm/sp adue cod. nome:banca	commissions	transfer, payment order	[727, 471, 598]	25.7%
603	liq. inter. ant.f competen	interests/charges	commissions	[202, 12, 70]	46.4%
704	bonifico a vostro favore bonifico sepa da banca popolare di puglia e basilica t	transfer, credit	commissions	[719, 23, 37]	52.9%
706	mav int.	external services	tax	[107, 105, 55]	55.6%
719	bonifico a vostro favore bonifico sepa da banca pop. alto adige per	transfer, credit	commissions	[704, 23, 37]	51.7%

TABLE 6: OPTICS - Error list with outliers.

Test ID	Test Description	Real Label	Assigned Label	Top 3 ID	Average Distance
84	bonifico a vs.fav. saldo ft 7 del 04.04.25 cro: bonifico n. disposto	transfer, credit	transfer, charge	[2, 89, 204]	61.2%
202	liq. inter. ant.f competen	interests/charges	commissions	[12, 70, 11]	69.9%
603	liq. inter. ant.f competen	interests/charges	commissions	[12, 70, 11]	69.9%
704	bonifico a vostro favore bonifico sepa da banca popolare di puglia e basilica t	transfer, credit	commissions	[23, 37, 9]	65.0%
706	mav int.	external services	tax	[107, 105, 55]	55.6%
719	bonifico a vostro favore bonifico sepa da banca pop. alto adige per	transfer, credit	commissions	[23, 37, 9]	63.2%

TABLE 7: OPTICS - Error list without outliers.

smaller average distance obtained with simple k-NN can be easily explained: there is wider number of points to select from, while, after sampling, the set of available points in the training set is reduced.

Evidence shows that classification errors occur more frequently when the average distance to the nearest neighbor is higher, hence it is possible to select a threshold marking as doubtful the descriptions with the average distance higher than that.

The effect of including outliers in the training set is exemplified by test ID 472 and 601 in Table 6, When outliers are included they are wrongly classified and their average distance to the nearest neighbors is less then half the value of standard errors. Once outliers are excluded as in Table 7, these two test ID are correctly classified and the average distance to the nearest neighbors is everywhere above 55%, as the only available neighbors are those belonging to well-defined groups, thus significantly worsening their average distance.

This evidence suggests an important development: it would suffice to introduce a threshold for the acceptable NN average distance, to isolate classifications to be considered "doubtful" to be checked manually or to be confirmed with a rule-based approach.

Conclusions

To overcome the limitations of the rigid and time-consuming rule-based classification of bank transactions, this section presented a Machine Learning approach, based on the

K-Nearest Neighbors algorithm. The model relies on the normalized Levenshtein distance enhanced with text pre-processing techniques, such as converting the text to lowercase, eliminating irrelevant sequences of digits or text and truncating long descriptions. With time, the size of the training set can increase significantly. To mitigate computational effort and scalability issues of the k-NN approach with large datasets, Hierarchical and OPTICS clustering methodologies were tested focusing on dataset reduction. After clustering transactions, representative points from each cluster were selected to form the new training set, reducing the original dataset size by 5-6 folds. The results obtained were very promising and can be summarized as follows:

1. The test performed using the full dataset, with "leave-one-out" method, achieved a surprising performance of 99% (6 transactions misclassified out of 834).
2. The test on the reduced dataset shows performances in line with those of the full dataset, also when excluding outliers.
3. The average distance of the tested descriptions to their NN can be used to provide a confidence measure for the assigned class and to define a threshold identifying doubtful classifications on which to concentrate direct controls.

In summary, the classification performances obtained with both clustering approaches are significative and consistent with those obtained using the entire training set, confirming that the methodology is scalable and reliable also with large training datasets. In addition, the average distance from the nearest neighbors, calculated for each test descriptions, provides a measure of the confidence of the assignment, suggesting which of them should be submitted to a direct manual check.

Classifying Transaction with LLM

In addition to the previous classical methods, this chapter considers modern LLM techniques for transaction classification. Unlike classical methods, advanced Natural Language Processing (NLP) algorithms can capture contextual meaning instead of focusing on isolated words or symbols. This may be useful when applied to the description of financial transactions, where the order of keywords may be relevant for the classification.

The vector representations generated by using **contextual embedding methods** adapt dynamically to their relative meanings, based on the context where they are inserted. For example, the word "bank" is represented differently in "river bank" than in "money bank." This represents a substantial improvement over static embedding approaches, which assign a fixed vector to each word, regardless of its meaning, thereby failing to capture polysemy and contextual nuance.

A major architectural breakthrough occurred with Google introducing the **Transformer** model. Transformers eliminate the need for sequential token-by-token processing, enabling **parallel computation** across all input positions. The innovation point is the **self-attention mechanism**, which allows to directly model relationships between any pair of tokens, regardless of their distance in the text. [14] This improves both computational efficiency and its ability to represent global context. The Transformer architecture has become the foundation for nearly all state-of-the-art NLP systems, in particular after introducing the Universal Language Model Fine-tuning (ULMFiT), an effective learning method that can be applied to any NLP task, which significantly outperforms the state-of-the-art on text classification tasks, reducing the error by 18-24%. [8] The Transformer architecture includes

three variants: encoder-only, decoder-only, and encoder-decoder models, all built on the same attention-based foundation.

- *Encoder - models* are designed to improve text understanding and representation and are well-suited for tasks like classification;
- *Decoder - models* (e.g., GPT, developed by OpenAI) generate text autoregressively, making them ideal for content generation;
- *Encoder - decoder* models combine both components and are effective for sequence-to-sequence tasks such as translation and summarization.

This work is focused on the BERT model, particularly appropriate for classification task [6]².

BERT consists of many pre-trained transformer layers, and task-specific adaptation can be achieved by training the parameters of the final layer to a specific sample.

A challenge in applying modern NLP methods to this study lies in the limited number of training samples, approximately 850, insufficient to provide a robust training to an advanced deep learning model, with millions – or even billions – of parameters. While traditional string-matching techniques, such as the **Levenshtein distance combined with k-Nearest Neighbors (KNN)**, require less extensive training sets as they rely solely on character-level similarity, ignoring linguistic meaning. On the other hand, Transformer networks require extensive corpora to generalize test effectively, consequently it is necessary to adopt frameworks to enhance procedure to function well in low-data environment. In additions, bank descriptions are typically short, loosely structured and often contain misspellings, abbreviations, or domain-specific jargon. Characteristics difficult to model effectively without substantial amounts of labeled data. For this reason no pretraining was performed and the raw BERT output was used instead. In addition, BERT was combined with a more traditional **Bag-of-Words (BoW)** representations. Pretrained language models already encode rich semantic information learned from large-scale corpora, enabling them to perform effectively even when fine-tuned on relatively small datasets. When coupled with simpler lexical representations, they provide a balanced framework capable of delivering reliable classification performance under data-scarce conditions.

The analysis is outlined in the following paragraphs by presenting:

- A. An outline of **text Tokenization**;
- B. The **spaCy environment**;
- C. The Bag-of-Words representation and **the BERT embeddings**, including a measure of **the confidence level** of classifications in the LLM environment;
- D. The test **results and conclusions**.

Tokenization

Tokenization is the process of segmenting a text string - such as a sentence or a paragraph into smaller, meaningful units called *tokens*. These units may correspond to complete words, word fragments (e.g., prefixes, suffixes), frequently recurring character sequences, individual characters or, less commonly, multi-word expressions. Serving as the fundamental units

²BERT is developed by Google and widely adopted in the industry. It is a state-of-the-art language representation model trained on large multilingual corpora, including Italian, using self-supervised objectives such as masked language modeling and sentence relationship prediction.

for computational models, tokens enable downstream machine learning systems to analyze and manipulate text in a structured manner [10].

To represent sentences as ordered sequences of tokens, early neural approaches introduced the use of encoders which are shown to learn a semantically and syntactically meaningful representation. Architectures such as **Recurrent Neural Networks (RNNs)** – including more advanced variants like **Long Short-Term Memory (LSTM)** networks and **Gated Recurrent Units (GRUs)** – process text sequentially, one token at a time by updating a **hidden state** that summarizes processed information [5].

This mechanism enables the model to capture word order and short-medium-range dependencies of tokens within a sentence, however, they tend to be slow to train and often struggle with very long-range dependencies due to issues such as vanishing gradients. Several established algorithms enable models to cope effectively with rare or unseen lexical items while maintaining semantic coherence. **WordPiece**, for instance, became prominent through its adoption in BERT as it provides a balanced trade-off between vocabulary size and linguistic flexibility [6].

Handling alphanumerical sequences, contained in transaction descriptions, may carry information relevant for classification. BERT handles such inputs through its sub-word tokenization process, which decomposes numeric and alphanumerical strings into smaller units and stores them in its vocabulary. These components are then embedded and contextualized, allowing the model to capture recurring structural patterns and incorporate them into the resulting embeddings. Through the construction of a consistent vocabulary – comprising all unique tokens appearing within a dataset – tokenization ensures standardized treatment of linguistic units across inputs.

In contrast, the Bag-of-Words approach, also adopted in the analysis, typically relies on standard tokenizers such as those provided by spaCy, which split text into whole words disregarding context. As a result, BERT captures semantic and contextual relationships, whereas Bag-of-Words represents text as simple word counts with no contextual awareness. One of BERT's core strengths is its ability to handle noisy and unstructured text inputs. By feeding resulting token sequences into ELMo and BERT, NPLs ability to recognize similarity between words, even with limited occurrences in the training data, is strongly improved, as contextual embeddings capture fine-grained language patterns. [11] Through its sub-word tokenization mechanism, BERT breaks unknown or distorted words into smaller units that become part of its learned vocabulary. This means that, even misspelled terminology or compound domain terms, can still meaningfully contribute to the final embedding. For this reason BERT is especially suitable to model bank transaction descriptions where numeric and mixed-format tokens appear frequently and carry domain-specific meaning, like the descriptions analyzed in this work.

The SpaCy Environment

One of the most established libraries for natural language processing is **spaCy**, which offers robust and linguistically informed tokenization, designed to efficiently split text into meaningful units. In our case spaCy is used to tokenize descriptions within the Bag-of-Words framework, since BERT uses its own algorithm (WordPiece) not suitable for the current purpose. In particular, spaCy provides a flexible framework enabling users to extend or customize tokenization rules.

SpaCy's tokenization process is primarily **rule-based** and can be enriched with predefined *special cases* associated with the selected language model, Italian in this case. These rules manage how the text is segmented into tokens while accounting for language-specific

patterns, such as abbreviations, clitic attachments and punctuation conventions. It handles abbreviations through **language-specific tokenization rules** and **special-case exceptions**, rather than through a fixed list of abbreviations. In the current work the following rules were applied:

- A. **Tokenization Rules and Exceptions** – SpaCy's tokenizer includes predefined *special cases* that treat certain sub-strings, such as common abbreviations, as single tokens. For instance, "SRL" and "SPA" can be written in different ways like "S.p.A" or "S p A ". Default tokenization algorithm may treat this as separate symbols ('s', 'p', 'a') and miss relevant information. To ensure consistent tokenization of these abbreviations, a special rule is added.
- B. **Format Sensitivity** – the original word format (upper/ lower case letters) is *preserved* during tokenization and the generation of the vector embedding. This allows the model to distinguish between upper and lower-case formats of the same word, which can provide further meanings for the following classification tasks. It should be noted that in Levenshtein distance approach only lowercase words were used to avoid mismatching, whereas BERT embeddings captures this information.

The Bag-of-Words Representation and the BERT Embedding

The *Bag-of-Words* (BoW) model is a foundational method in natural language processing that represents a text document as a numerical vector based on the frequency of its words. [15] The model treats the text as a "bag" of individual terms, thereby disregarding grammar, syntax, and word order, assuming that, *"the meaning of a document can be inferred from word counts alone."*

The construction of a BoW representation begins with the creation of a **vocabulary** derived from the entire corpus under analysis. This vocabulary consists of all *unique* tokens or, in some cases, sequences of tokens (n-grams), appearing across the dataset. Each unique token or n-gram is then treated as a distinct **feature** in the resulting vector space. Consequently, every document is mapped onto a vector, where each element corresponds to the frequency with which the associated token occurs.

Because different inflected forms of a word (e.g., *work, working, worker*) may refer to the same underlying concept, many workflows apply **lemmatization** prior to frequency counting, a process that reduces words to their canonical form. As noted in computational linguistics, *"lemmatization consolidates morphological variants to ensure consistent representation of concepts"*. The spaCy library provides efficient and linguistically informed tools to perform such lemmatization. However lemmatization was not considered in this study because in transactions different words referring to the same concept may carry relevant information which should not be lost.

For each individual text segment (e.g., a product description), the BoW procedure computes the frequency of every vocabulary 'word' within that segment. The resulting vector is typically **sparse**, meaning that most values are zero, since any single text contains only a fraction of the full vocabulary. A sparse representation is a characteristic feature of BoW-based encoding and reflects the high dimensionality and lexical diversity of the selected text. The following paragraphs shows how to create advanced contextual embeddings that can capture semantic and contextual information.

The BERT Embedding

To capture the semantic and contextual information embedded in bank-transaction descriptions, this study employs a pretrained BERT Transformer exclusively as an **embedding**

generator, transforming each sparse transaction description into a dense, fixed-length vector appearing as a rich and informative feature representation.

The model's capacity to infer contextual meaning can be illustrated using the transaction description: "ACCR. BONIFICO EU COD. DISP.: 0124".

When processed by BERT, the output is a single dense vector that captures the latent relationships among the abbreviated and domain-specific terms. For instance, "ACCR." is a shortened form of *accredito*, while "BONIFICO" combined with the token "EU" suggests an association with *EU-area SEPA transfers*. The resulting embedding occupies a region of the vector space proximal to other transaction descriptions expressing semantically analogous operations, related to incoming SEPA transfers, such as:

- "accredito bonifico SEPA";
- "transfer received EU";
- "movimento bancario in entrata";
- "incoming payment code 1234".

Sentence classification is the most direct and intuitive applications of BERT, as illustrated by Jay Alammar in [1] In practice this simple embedding and fine tuning, leverages BERT's pre-trained linguistic knowledge and often yields strong performance with minimal specific engineering.

Enhanced Bag-of-Words Representation

Traditional approaches like Term Frequency-Inverse Document Frequency (TF-IDF) and Bag-of-Words, offer simple and computationally efficient means of representing term importance within a document, but they lack the ability to capture semantic relationships. Recent works have shown that these lexical representations can be enhanced through the integration of pretrained embeddings capable of modeling semantic and contextual information as in [4] where the BoW representation and BERT are mixed through a weighted feature-level strategy. Empirical results demonstrate that hybrid models achieve superior training performance and attains the highest test accuracy among the evaluated configurations.

In practice, in this study:

- The text was tokenized using spaCy, to ensure that all tokens are preserved, including punctuation, domain-specific terms, and alphanumeric strings;
- These were then converted into a BoW vector reflecting the raw frequency of each token in the description (i.e. without applying TF-IDF weighting or normalization);
- Similarly, transaction descriptions were also encoded using BERT to generate a deep semantic embedding;
- Finally, the two vectors were concatenated to create a combined feature vector leveraging both high-level contextual understanding and straightforward term-frequency information.

By leveraging both contextual and lexical cues, the hybrid model provides a more comprehensive and discriminative representation of textual content. This joint vectors represent the inputs for the classification task.

The selection of the classification models depends on several factors, in particular: data availability, separability of classes and a confidence interval of classifications.

- **Data availability** is a key driver for model selection. Given the relatively small size of the available sample, classical Machine Learning classification models are considered as they perform well both on small and large dataset. However, the framework is built to accommodate large amounts of data which allow to consider more advanced methods, particularly BERT embeddings for domain specific classification tasks.
- **The confidence level of classifications** requires to define a metric identifying poorly classified descriptions. Classifications with a low confidence score should be investigated manually.

Potentially, with a larger sample of data, ensemble framework could be used, i.e. run multiple models and average their predictions.

Measure the Confidence Level of Classifications

With the K-NN algorithm to identify a measure for the confidence level was straightforward and intuitive: the average Levenshtein distance to the closest neighbors, representing the degree of overlapping of the test description with the selected sentences. In the LLM framework however, the output of the classifications process consists of the a probability distribution Y of the different tags being assigned to the test description through a multinomial logistic regression;

$$Y = \begin{bmatrix} P(tag_1|x) \\ P(tag_2|x) \\ \dots \\ P(tag_n|x) \end{bmatrix} \quad \text{with} \quad \sum_i P(tag_i) = 1,$$

with the tag effectively assigned to the test description being that with the highest probability,

$$\text{tag}(x) = Y_i \quad \text{where} \quad P_i = \max\{P(tag_i|x)\}.$$

In this case the probability vector Y is given by the **Softmax** function 1, one of many functions mapping non-normalized scores onto probability vectors, but one particularly useful in IO models, where only the right prediction is relevant and not the transparency or accuracy with which the output is modeled.

$$Y_i(x) = \frac{\exp^{-a_i x_i}}{\sum \exp^{-a_i x_i}}. \quad (1)$$

Clearly, the confidence level of this assignment depends not only on its probability value, but also on how the remaining probability is spread across the other classes. One possible approach to measure uncertainty in the LLM world, is to look at the **predictive entropy** equation 2, representing the average amount of information within the predictive probability distribution:

$$E = - \sum_{i=1}^n p(tag_i) \log(p(tag_i)), \quad (2)$$

where the summation is over all possible tags that Y can assume. The predictive entropy reaches its lowest value when all classes are associated with equal probability, indicating complete uncertainty. Conversely, in case of a very confident classification, it reaches its highest value of near-0, when one class is predicted with near certainty (a probability of near-1).

Split	Above Threshold		Below Threshold		Whole Test Set			
	Points	Errors	Points	Errors	Accuracy	Accuracy	Correct Tests	Wrong Tests
1	194	0	15	3	80%	99%	0.91	0.47
2	198	0	11	1	91%	100%	0.91	0.66
3	197	0	12	1	92%	100%	0.91	0.63
4	197	0	12	0	100%	100%	0.91	-
5	199	0	10	1	90%	100%	0.92	0.66
6	200	0	9	1	89%	100%	0.91	0.58
7	195	0	14	0	100%	100%	0.90	-
8	191	0	18	8	56%	96%	0.91	0.39
9	201	0	8	2	75%	99%	0.92	0.40
10	196	0	13	1	92%	100%	0.91	0.69
11	199	0	10	1	90%	100%	0.91	0.22

TABLE 8: LLM results by iteration.

Test Results and Conclusions

Cross Validation

To obtain reliable evaluation results, cross-validation is applied by training and testing the classification model on multiple data splits. Leave-One-Out Cross-Validation (LOOCV) can be effective for small datasets and lightweight models as was described in Chapter 1. However, it becomes computationally impractical for more complex architectures with advance classification models. Here, instead, we use Stratified Cross Validation.

In **stratified cross-validation**, the dataset is partitioned preserving the relative proportion of each key variable between the training and testing folds. This means that each partition reflects the original class distribution, an essential requirement when dealing with imbalanced datasets or rare categories. The stratification constraint ensures that each fold is a representative sample of the full dataset with respect to the target variable. This reduces variance in performance estimates and avoids biased evaluations that can occur when certain classes are over or under-represented in the test sets. By maintaining consistent class proportions across folds, stratified cross-validation produces more stable and meaningful comparisons between different models.

The model is then trained and evaluated multiple times across different stratified folds, yielding a distribution of performance metrics, such as errors, accuracy and confidence level. This richer set of results increases statistical robustness and enables the detection of systematic mis-classification patterns recurring across folds. Categories distribution is not particularly concentrated in one or two classes. Moreover, errors in classification are usually encountered among the most frequent categories: "bonifico addebito and commissions". It implies that current model allows to separate more specific categories of transactions. Consequently, stratified cross-validation provides a more comprehensive and reliable assessment of a model's generalization capabilities than either a single train-test split or non-stratified cross-validation methods. Its effectiveness is particularly pronounced when working with datasets characterized by class imbalance, limited size, or heterogeneous label distributions.

Results and Conclusions

This research provides a comprehensive evaluation of automated systems for classifying bank transactions, a tedious but vital component in the reconciliation of invoices and

financial records. The study proves that the integration of AI and ML techniques offers a precise, consistent, and scalable solution for financial control. Both traditional distance-based methods and modern contextual models can achieve a significant accuracy (99% and above), comparable to manual rule-based systems, along three main dimensions. It should be noted that the model performs well even for transaction categories with a small number of training samples; therefore, the accuracy metric is considered a reliable indicator of overall performance.

- **Scalability and Efficiency:** the first part of the work demonstrates that the **K-Nearest Neighbors (k-NN)** approach, using Levenshtein distance, is highly effective for short text segments. Furthermore, the application of **HAC and OPTICS clustering** successfully help reducing the training dataset size by **5 to 6 times** without compromising performance, providing a clear path for managing scalability as transaction volume grows.
- **Semantic Depth:** the second part reveals that **Large Language Models (LLMs)** like BERT can interpret the contextual meaning and domain-specific jargon of financial data, even when training samples are limited. The results from the LLM-based approach, reported in Table 8, demonstrate that integrating BERT embeddings with Bag-of-Words (BoW) representations successfully overcomes the limitations of small, unstructured training datasets in the financial contexts. By combining high-level contextual understanding with straightforward term-frequency information, the hybrid model achieved an average accuracy exceeding 99% across eleven iterations of stratified cross-validation.
- **Practical Implementation:** a significant outcome for both methodologies is the ability to generate **confidence metrics**, which allow organizations to focus manual efforts exclusively on "doubtful" cases. **Predictive entropy** was selected as a measure of the confidence level for classifications in the LLM environment. A key finding of this work, was the effectiveness of the 70% confidence threshold, above which zero classifications errors were found, hence serving as a reliable indicator for triggering manual controls for the few classifications obtaining a smaller confidence level.

This confirms that advanced NLP techniques can be adopted also for low-data environments to capture the nuanced meaning of bank descriptions, ensuring a robust and trustworthy automated classification process. 

References

- [1] **Alammar, J.** *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)*, Hands-On Large Language Models, 2018.
- [2] **Almar, J. and Garcia, J.** *The Impact of Artificial Intelligence and Machine Learning on Management Accounting*. International Journal of Multidisciplinary Research and Publications, ISSN 2581-6187, pp. 114-121, 2025.
- [3] **Ankerst, M. et al.** *Optics: ordering points to identify the clustering structure*. ACM SIGMOD Record, Vol. 28, issue 2, pp. 49-60, 1999.
- [4] **Aprilio, P. et al.** *Hybrid Feature Combination of TF-IDF and BERT for Enhanced Information Retrieval Accuracy*. Jurnal Informatika dan Sains, June 2025.
- [5] **Cho, K. et al.** *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014.
- [6] **Devlin, J. et al.** *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. October 2018
- [7] **Fieberg, C. et al.** *Machine Learning in Accounting Research*. Springer, pp. 105-124, September 2022.
- [8] **Howard, J. and Ruder, S.** *Universal Language Model Fine-tuning for Text Classification*. January 2018.
- [9] **Munoz, J., Jalili M. and Tafakori, L.** *Hierarchical classification for account code suggestion*. Knowledge-Based Systems, Volume 251, September 2022.
- [10] **Phan, C. and Singh, I.** *Understanding WordPiece Tokenization: An Approach to Subword Units*. Collection of Machine Learning Interview Preparation materials, 2026.
- [11] **Salunke, A.** *Enhancing Contextual Understanding in NLP: A Subword Tokenization Approach with ELMo and BERT*. June 2023.
- [12] **Sebastiani, P. and Perls, T.** *Detection of significant groups in hierarchical clustering by resampling*. Frontiers in genetics, Vol. 7, pp. 144, 2016.
- [13] **Tootian, P.** *The Impact of Artificial Intelligence Application on the Accounting Profession*. 2023
- [14] **Vaswani, A. and et al.** *Attention Is All You Need*. June 2017.
- [15] **Zhang, Y. et al.** *Understanding bag-of-words model: a statistical framework*. International Journal of Machine Learning and Cybernetics, August 2010.

Iason is an international firm that consults Financial Institutions on Risk Management.

Iason is a leader in quantitative analysis and advanced risk methodology, offering a unique mix of know-how and expertise on the pricing of complex financial products and the management of financial, credit and liquidity risks. In addition Iason provides a suite of essential solutions to meet the fundamental needs of Financial Institutions.



**ESSENTIAL SERVICES FOR
FINANCIAL INSTITUTIONS**