**TUM Data Innovation Lab**

Munich Data Science Institute (MDSI)

Technical University of Munich

&

**Munich RE**

Final report of project:

# Automatic Mapping of Incoming Payments to Business Partners

|  |  |
|---|---|
| Authors | Xu He, Helena Franta, Joshwa Georgekutty, Erblina Jakupi |
| Mentors | Jennifer Hahn, Bernd Heilsberger, Manuel Kuhlin, Privalika Janardhan, Martin Baldus, Jack Tan, Hesiona Murati |
| TUM Mentor | Dr. Alessandro Scagliotti |
| Project lead | Dr. Ricardo Acevedo Cabra (MDSI) |
| Supervisor | Prof. Dr. Massimo Fornasier (MDSI) |

Jul 2024

## Abstract

Munich Re currently aims to fully automate the internal cash clearing process in accounting. To this end, a necessary preliminary step is to map each incoming payment to its corresponding Business Partner. Automating this process is also a foundation for future automation of the booking process. Inconsistencies in reference texts and the high amount of transactions make this task hard to be maintained on a semi-automated level, as is done currently. We suggest a combination of Direct Mapping and Machine Learning Models to map incoming payments.

First of all, we performed Exploratory Data Analysis and necessary preprocessing on the historical SAP data provided by Munich Re. In a second step, we explored several modeling approaches and explain how to efficiently make use of the underlying text data in a Machine Learning context. We employ Keyword Extractors and Fuzzy Matching Techniques to extract the most information out of the Reference Texts. To optimise their computation time, fine-tuning of the parameters is necessary. However, the majority of the Reference Texts do not contain a unique identifier. While exploring different Machine Learning Methods, we encounter the problem of handling a very imbalanced dataset as well as an abundance of classes. In the end, we find a Decision Tree to give us the highest accuracy. Our final solution design relies on a combination of different approaches, providing extra flexibility in usage and retraining processes due to its modularized design.

# Contents

# 1 Introduction

## 1.1 Motivation

Munich Re, the world's largest reinsurer with over 40,000 employees and annual revenues of around 60 billion Euros [4], currently faces significant challenges in automating their cash clearing process. *Cash clearing* in business refers to the process by which transactions involving cash (or cash equivalents) are settled and reconciled. This involves verifying that funds have been received and properly accounted for, and ensuring that all transactions are accurately recorded in the company's financial records. The cash clearing process helps to prevent discrepancies, fraud, and errors in financial statements. For cash clearing it is necessary to first allocate each payment in its payment lot, which can only be done if the Business Partners (BuPas) of the respective payments were correctly identified. Automatically identifying respective BuPas therefore is a necessary preliminary step for automatic cash clearing.

The existing mapping methods are primarily experience-based, and this process is becoming increasingly problematic due to anticipated knowledge loss from retirements and the error-prone nature of semi-automated, semi-manual processes, especially with a rising number of transactions. The core objective of our project is to automate the mapping of incoming payments to their respective BuPas, ensuring transparency and reliable financial transactions in compliance with IFRS 17, while increasing shareholder trust in the enterprise.

## 1.2 Stakeholders

The key stakeholders for Munich Re's project to automate the mapping of incoming payments to their respective BuPas include employees, management, and executives, who are directly involved in the process and will benefit from reduced manual workload and errors. Compliance officers, responsible for ensuring regulatory adherence, will benefit from enhanced accuracy and transparency in financial reporting. BuPas rely on accurate and timely payment mapping for their transactions and will benefit from a more reliable and efficient process. Each of these stakeholders has a critical role and stands to gain significantly from the successful implementation of this automation project.

## 1.3 Business Benefits

Automating the mapping of incoming payments at Munich Re offers numerous business benefits. Firstly, it significantly increases operational efficiency by reducing the time and effort required for manual maintenance and transaction processing. This automation minimizes the risk of errors associated with manual and semi-automated processes, ensuring more accurate financial transactions and reporting. Additionally, it mitigates the impact of knowledge loss due to retirements, as the system no longer relies on individual experience. Regulatory compliance is enhanced as accurate and transparent financial transactions help maintain adherence to IFRS 17 and other requirements.

### 1.4 Scope of systems

As Munich Re has companies all over the world, a standardized way of documenting each financial transaction is needed. Munich Re's General Ledger is called Global Template (GT), which poses a possibility to create bookings in a uniform way globally. It is useful to harmonize Business Processes to ensure consistency and efficiency across the organization. In GT one can find transactions of all bank accounts of Munich Re and it is a direct link to the different bank accounts. Payments are usually uploaded automatically or uploaded in batches from the bank account statement. These payments are not yet allocated in its payment lot. To be able to allocate the transactions, additional information from the individual BuPas is necessary (Statement of Accounts). Statement of Accounts include the accounted figures of the Reinsurance Business (e.g. premiums and losses) and are booked in SAP FS-RI (Financial Services - Re-Insurance). The Cash Clearing can then be done in SAP FS-CD (Collections and Disbursements) with the created payment lot and the booked figures from the Statement of Accounts.

What therefore is needed in order to create historic training data, is to find a mapping between GT entries (stemming directly from the bank accounts) and the FS-CD bookings which are already allocated to a Payment Lot. In the end, the goal is to allocate each payment in the bank accounts to their respective Payment Lot. In our project, we aim to directly find a matching BuPa for *new* entries in GT, and therefore to match new incoming payments automatically. To this end, we will train the matching on historical entries out of GT.

# 2 Statistical Data Exploration

We follow a typical Data Mining Process as described by Wirth and Hipp in [8]. We first started with business understanding, and in this chapter we will now focus on data understanding and its preparation. Afterwards, in chapter three we will explore modeling possibilities, evaluate them and finally deploy the best one.

## 2.1 Data Acquisition

The datasets provided to us are extracted out of SAP. Each contains historical data going back to 01.01.2010. Out of GT there are 12 extracted data tables for each separate company (e.g. Munich Re Australia) from this time frame. First, we create the historical mapping of each transaction to their respective BuPas. We can match historic FS-CD entries with GT entries via their Reference Key, clearing document number and the respective amount of the transaction. Using this method we are able to obtain a Basic Population or historical dataset containing over 20,000 non-FS-CD entries connected to their correct BuPa. This dataset is used as our historic training dataset. Additionally, we were provided with one list showcasing all possible (active as well as inactive) BuPas and their corresponding 6-digit BuPa IDs. Furthermore, we received a list of all Treatys of Munich Re. The 7-digit Treaty ID can be used to identify the respective BuPa.

## 2.2 Data Cleaning and Validation

Data cleaning and validation are critical components in any Machine Learning project. They ensure that the data used for training and testing models is of high quality, which directly impacts the performance, reliability, and interpretability of the models. The data cleaning process involved several steps to enhance the quality and usability of the data-set.

**Reference Keys Standardization:** Reference keys were found to contain leading and trailing 0s, as well as special characters such as slashes ('/') and asterisks ('*'). To ensure consistency and ease of analysis, these irregularities were addressed by standardizing the format of reference keys.

**Removal of 'M01' Ending for Manual Bookings:** Within the reference key column, instances of the ending 'M01' were identified, indicating manual bookings. To maintain uniformity and clarity, this suffix was removed from relevant reference keys.

**Reviewing Relevant Columns:** A thorough examination of the data-set was performed to assess the significance of each column in relation to the objectives of the analysis. The aim was to identify columns containing essential information that directly contribute to the analysis and decision-making process.

## 2.3 Data Preprocessing

Text Data needs special preprocessing to reduce inconsistencies and standardise the text across the datasets. We applied the following techniques to the Reference Text as well as the BuPa Names.

**Lower Casing:** The entire text is converted to lowercase for uniformity.

**Tokenizing Text:** Tokenizing means the process of breaking down unstructured text into smaller units. In our case, this means separating words from each other whereas we do not split up any strings of numbers.

**Removing Punctuation:** Punctuation may increase noise in text data and deleting it helps to reduce the complexity of our models. To focus only on relevant information, we deleted punctuation (such as "," , ".") from the text column.

**Deleting Stop Words:** Stop Words are words that appear frequently in text, but do not add any value to its meaning. Common examples are conjunctions (e.g. "and", "or") and articles ("the", "a"). These words again only add noise and their removal helps to reduce the complexity of our model. Given our reinsurance and payment-focused business case, we also ran a special analysis to identify domain-specific stop words in our available data. We first filtered for the most common words available. Next, we selected 60 words out of that list to be removed from the reference text that do not help to identify an individual BuPa. Specific stop words we found are e.g. "payment", "transfer" , "life" and related abbreviations, e.g. “pt” or “pymt”.

We found Text Preprocessing to be effective as it in fact *decreased the Levenshtein distance* between preprocessed Text and actual BuPa Name compared to the original Reference Text and the BuPa Name. Indeed, it decreased Levenshtein distance *on average* by 22. We further explain the Levenshtein Distance more in detail in Chapter 3. To use Text data as input for Machine Learning models further special preprocessing is necessary which we described here.

## 2.4 Data Exploration

Data exploration, involves investigating, summarizing, and visualizing data to uncover patterns, anomalies, key variables, and the underlying structure. After pre-processing the historical mapped data, the exploratory analysis was conducted, focusing on two primary features: the reference text and the entry date, which corresponds to the actual date the transaction was recorded in the General Ledger (GT). The primary objective was to identify any direct mapping pattern using the reference text and to discern behavioral patterns within the entry date column.

### 2.4.1 Reference Text Exploration Analysis

The most promising feature to help identify the BuPa in our dataset seems to be the Reference Text as it comes closest to being a unique identifier. By cleaning the Text and removing stop words we could further improve its importance.

We want to note that the Reference Text column is limited to 50 characters by SAP constraints, leading to Truncation of the original References. It is possible that some BuPas have been mentioned in the truncated part. The analysis of the text field revealed distinct patterns within the Reference Text column. Notably, the column contains entries identified as NaN (Not a Number), constituting approximately `4.86%` of the total entries. A common pattern observed involves entries prefixed with the word “from”, potentially indicating a reference to a BuPa, e.g. “123456789 from [**BuPa Name**]”. `10%` of the total entries contain the keyword “from”.

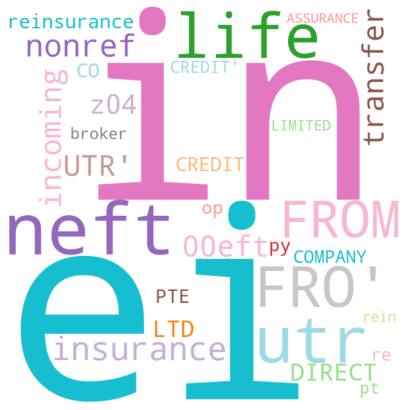


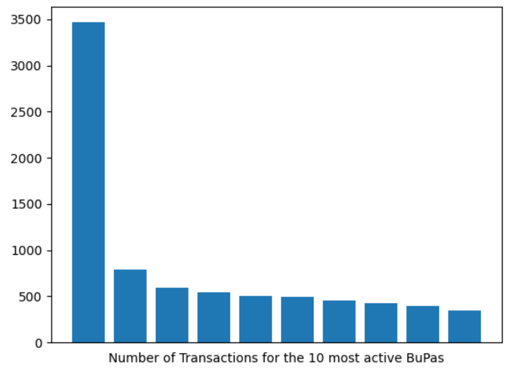
Figure 1: Most common words in the unprocessed Text References.

### 2.4.2 Entry Date Exploration Analysis

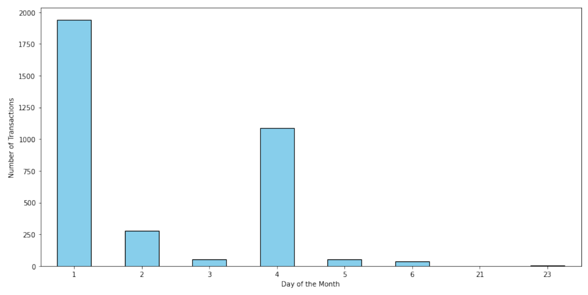
For the distinct entities identified as BuPas in the training data, only around 82% have conducted multiple transactions. The number of transactions for the most active BuPas can be seen in Figure 2a. This indicates that a significant majority of BuPas engage in repeated transactions, highlighting a pattern of ongoing interactions within the data set.

The exploration analysis reveals a systematic behavior in the scheduling of transactions. Specifically, transactions tend to occur consistently at the beginning of each month. This regularity suggests a structured and predictable transaction pattern, likely driven by monthly financial cycles, recurring billing schedules, or planned transfers.

To analyze the transaction patterns, a bar plot was generated to show the distribution of transaction days within each month. It illustrates the number of transactions for each day of the month. This visualization demonstrates a clear concentration of transactions occurring at the beginning of each month, confirming the systematic behavior in transaction scheduling.



(a) Number of Transactions for the Most Active BuPas



(b) Distribution of Transaction Days for the BuPa with the Highest Number of Transactions

Figure 2: Transaction History Overview

## 2.5 Challenges and Edge Case

### 2.5.1 Data Challenges

**Absence of Standard Reference Texts:** The absence of standardized Reference Texts poses a challenge in accurately classifying and interpreting data entries. Without clear reference points, ensuring consistency in data analysis becomes difficult.

**Uninformative Reference Texts:** Certain Reference Texts may lack sufficient information or context, making it challenging to extract meaningful insights from the data. This issue can lead to ambiguity and inaccuracies in data analysis and decision-making processes. As the Reference Texts have a character limit of 50 characters, in some cases the BuPa name may have been truncated.

**Rare or Infrequent Transactions:** Rare or infrequent transactions pose a challenge in terms of behavioral analysis and forecasting. About 18% of BuPas only have a single one-time transaction in our dataset, which significantly impacts the balance of our training data.

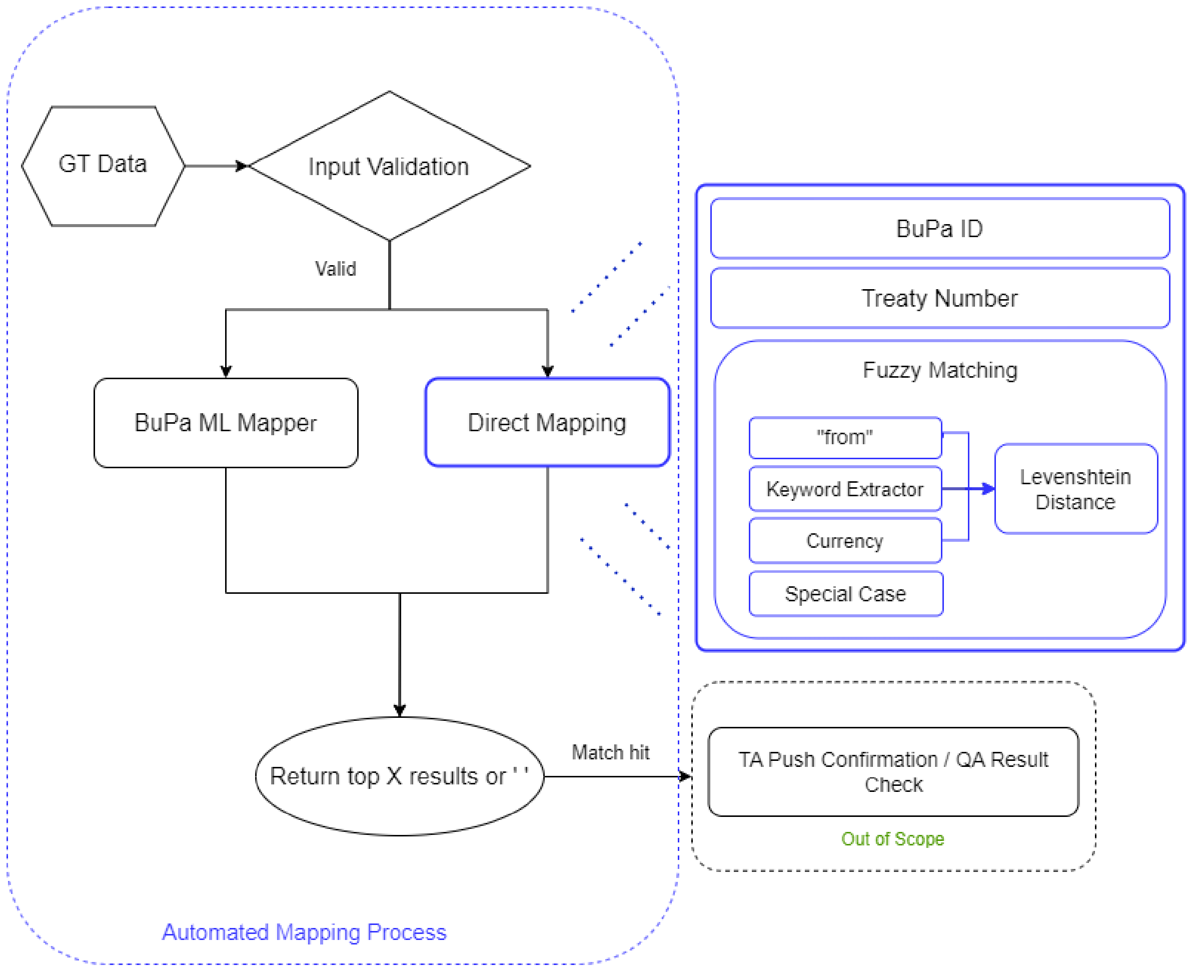
### 2.5.2 Edge Case

**Modification of Business Partner name:** Changes in Business Partner names resulting from mergers or acquisitions can impact data consistency. Historical data associated with previous BuPa names may need reconciliation or updating.

# 3 Business Solution Design and Development

## 3.1 Solution Architecture

We suggest a solution design that first tries to complete any easy matches we may find. If a direct match in the Text Reference (via BuPa ID, Treaty ID or Fuzzy Matching) is found during Direct Mapping, the process proceeds to directly return the BuPa. If, however, no direct match can be found, this part returns a list of possible Matches by Fuzzy Matching and the data is passed to a more intricate Machine Learning Model for further analysis. Based on its output, the model may return in the end a single hit or a list of the most likely matches. Additionally, Technical Accountants (TA) can in the future be involved in the process by confirming the suggested matches.



## 3.2 Identifying influencing factors

In this phase of the project, careful consideration was given to selecting the features or columns that are expected to have a significant influence on the outcome or target variable (BuPa ID). By identifying and incorporating these factors into the analysis, we aim to uncover meaningful patterns and relationships that contribute to the overall predictive accuracy and effectiveness of the model.

**Selected input features:**

- **Company Code:** Indicates for which location this entry is done.
- **G/L Account:** General Ledger (GL) accounts only include incoming payments (no outgoing payments). One Company Code can have multiple GL accounts, separated for Life and Non-Life business, different bank companies and so on.
- **Document Currency Key:** Original currency which was received by Munich Re.
- **Document Currency Value:** Original amount which was received by Munich Re.
- **Pre-processed Text:** Processed text which was sent by the cedent with the payment.
- **Entry Date:** Shows the date when that entry was created.
- **Local Dimension 1:** A local dimension value is available only for Singapore.

After preprocessing the data and excluding FSCD entries, the training dataset comprised over 20,000 entries. These entries were carefully curated to ensure that relevant features and the target variable were included, providing the model with sufficient information to learn and make accurate predictions.

## 3.3 Splitting Data for Cross Validation

To ensure the robustness and reliability of the Machine Learning model, cross-validation was employed using 10 subsets of the training dataset. This approach involves dividing the dataset into 10 equal parts, or folds, where each subset is used as both a training and validation set in turn.

By iteratively training and evaluating the model on different subsets of the data, we aim to assess its performance across multiple scenarios and mitigate the risk of bias or variance. This cross-validation technique helps validate the model's predictive ability and ensures that it generalizes well to unseen data.

## 3.4 Data Transformation

Transforming categorical features into numerical ones is essential for training Machine Learning models because most Machine Learning algorithms operate on numerical data. For this study, we identified the following categorical features: Company Code, G/L Account, Document Currency Key and Local Dimension 1. These features were encoded using *Label Encoding*, a method that assigns each category a unique integer based on its order. Label encoding preserves the ordinal relationships within categorical data, which can be crucial for certain algorithms that can leverage this information during training.

We also tested *One-Hot Encoding*. One-Hot Encoding creates dummy features for every categorical value found in the column, except for one. Each new feature then is a binary variable, indicating whether this value was present in the column before. We did not use it in the end, as it additionally increased Dimensionality and could not preserve ordinal relations.

Additionally, in the transformation process, normalization was applied to handle features such as Document Currency Key, which were already numerical but required scaling for consistency across different magnitudes. Similarly, Entry Date was also normalized to convert temporal data into a numerical format suitable for Machine Learning algorithms. Normalization ensures that these features are standardized and contribute effectively to the model's training process by maintaining relative relationships and scaling them appropriately within the dataset.

**Preparing the text data for Training** For the Reference Text feature, we tried using a very similar approach to One-Hot Encoding called *Count Vectorizer*. For every word found, a new column is created. If the word is present in the reference text, its count will be the value of the respective column. The idea becomes clearer with the following example:

| Reference Text |
|---|
| pymt from abc |
| quarterly transfer abc |

| from | pymt | quarterly | transfer | abc |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |

One can already see this method will result in a wide and sparse matrix consisting of many features. At a certain point, the more features there are, they will not necessarily add information of value to a Machine Learning model. It also increases the tendency of models to overfit, e.g. the training data will be memorised and unseen data is not classified correctly.

## 3.5 Direct Mapping

### 3.5.1 One-to-One Matches

In this first approach we want to catch all easy cases where one can *directly* find the respective BuPa by essentially reading it off. The BuPa can be directly identified by

- its 6-Digit ID
- a 7-digit Treaty ID that identifies the respective Treaty where the BuPa can be looked up.

For the IDs we use RegEx (Regular Expression) to extract possible strings out of the Reference Text field that contain the specific amount of numbers. These hits are then looked up in the corresponding tables. If there is an exact match (i.e two IDs are identical) we will assign the payment to the found BuPa. We discuss the results of this one-to-one Mapping in Chapter 4.

### 3.5.2 Fuzzy Matching

In contrast to an exact match, *fuzzy* matching will also match strings that e.g. differ in only one character. One way to measure the difference between two strings is the *Levenshtein Distance.*

**Levenshtein Distance:** The Levenshtein Distance measures the difference between two strings of *different lengths.* Other Text distance measurements, such as e.g. the *Hamming Distance* instead require the strings to have equal lengths. Levenshtein Distance counts the number of substitutions, deletions or insertions of characters that is necessary to transform one string into the other [1]. Compare the two strings "payment" and "pymt": In the latter string, one needs to insert an "a" as well as "en". Therefore the Levenshtein Distance here is 3. We can see that the *higher* the Levenshtein Distance, the more different the strings.

One important thing to note is that the Reference Text Field is limited to 50 characters (because of SAP constraints). For the Levenshtein Distance this means that any distance is always bounded from above by 50. Therefore, any Levenshtein Distance above 30 already refers to two quite unrecognizable strings. This also highlights again the importance of preprocessing the text where we could achieve an average decrease of 22 (with 50 as a maximum) in the Levenshtein Distance.

### 3.5.3 Fuzzy Matching with Keyword Extractors

Apart from identifying the BuPa via its ID, it is also possible to identify it via its Name spelled out in the Reference Text. Trying to match the entire Reference Text to a Name, however, is inefficient and we therefore improved the performance of matching entire Names of the BuPa present in the Reference Text by exploiting Keyword Extractors. The idea of a Keyword Extractor is to select the most characteristic words (*keywords*) in a given text. In the best case, the Keyword Extractor would extract the exact name of the BuPa in the Reference Text column. This makes it very easy to find the corresponding BuPa on the list of all BuPas. One major shortcoming of this approach is that it *only* ever *can* work if the BuPa is mentioned by its name in the Reference Text. If that however, is indeed the case, Fuzzy Matching the keyword will be successful.

We have compared three different Keyword Extractors.

**SpaCy:** SpaCy [7] is the standard library for Natural Language Processing (NLP) in Python. SpaCy relies on trained functions, and is trained on large dictionaries. Based on NLP libraries, it operates by extracting linguistic features of a text based on the chosen language model i.e. English, German, Spanish etc. SpaCy is therefore tied to a specific language.

**YAKE:** YAKE is short for *Yet Another Keyword Extractor* [9]. It is an unsupervised approach that therefore does not rely on external dictionaries. Unlike SpaCy, YAKE does not have a pre-trained language model built in, however, the logic relies on statistically analyzing the frequency of terms and the distribution of terms across different parts of the text.

**KeyBERT:** KeyBERT on the other hand, is based upon the Large Language Model BERT (Bidirectional Encoder Representations from Transformers) [2]. It leverages the Deep Learning capabilities of BERT to capture the contextual relationships between words. It has different pre-trained deep-learning models one can use. KeyBert provides flexibility and excels at capturing contextual and semantic meaning.

Table 1 below shows a detailed explanation of their differences in performance, complexity and their approach.

| **Extractor** | **SpaCy** | **YAKE** | **KeyBERT** |
|---|---|---|---|
| **Approach** | Rule-based and statistical techniques for keyword extraction. | Statistical method focusing on term frequency and positional importance. | Combines BERT embeddings with clustering algorithms to extract keywords. |
| **Language Model** | Uses pre-trained language models for linguistic analysis. | Does not rely on pre-trained language models; uses a statistical approach. | Uses BERT, a state-of-the-art transformer-based language model. |
| **Customization** | High; can be customized with different pipelines and models. | Moderate; primarily parameter tuning for statistical measures. | Moderate to high; allows fine-tuning of BERT and clustering parameters. |
| **Complexity** | Requires understanding of NLP pipelines and models. | Involves understanding statistical keyword extraction principles. | Requires knowledge of BERT and clustering techniques. |
| **Performance** | Good and very fast | Effective. | Excellent but very slow. |
| **Dependency on Context** | Moderate; can capture context with dependency parsing. | Low; primarily based on statistical properties of terms. | High; captures contextual and semantic due to BERT embeddings. |
| **Ease of Use** | Integrated into the SpaCy library. | Easy to use with minimal setup. | Requires setup and understanding of BERT and clustering. |

Table 1: Comparison of SpaCy, YAKE, and KeyBERT for keyword extraction

All of these extractors, extract not one, but several possible matches, that are ranked by their score. Each of these extractors, have a different approach in defining the score which means each word will be given a score based on its importance either calculated on frequency for Yake, or semantic meaning for KeyBert.

Consider for example for the sentence " Students can apply for the TUM DI Lab every semester." Possible extracted results would be "TUM", "TUM DI", "TUM DI Lab","students", "apply". In this example, one can see one shortcoming of using Keyword Extractors to extract Names: the length of the Name is not clear. Most of the time, entity names consist of several words. We would like "TUM DI Lab" to be extracted *entirely* instead of only getting one part of that name. In fact, the majority of the underlying BuPa names consist of multiple words (think of combinations of e.g. "limited", "company", "insurance").

To avoid this, a lot of fine-tuning is necessary to find the optimal parameters for each Keyword Extractor. Fine-tuning will reduce the number of possible keywords. For example the base model for Spacy returns over 28,000 possible keywords for a short Reference Text containing only five words, whereas the base Model of YAKE only extracts 18 possible keywords for the same case. After fine-tuning YAKE, 6 possible keywords were extracted for the same case. This concludes that refining and choosing the right Keyword Extractor is a crucial step in improving Fuzzy Matching.

In our case, although KeyBert is the best in capturing semantic meaning of a text, the computation time it takes is two times slower than the one of Yake or SpaCy, making it not suitable for our final solution. To use SpaCy one needs to define the language model which is inefficient in our case as we have different languages in Reference Texts from different countries. We therefore chose YAKE as the most suitable technique in delivering effective Keyword Extraction and improving Fuzzy Matching performance.

### 3.5.4 Further Optimizations

To further optimize Fuzzy Matching, we leverage domain-specific characteristics of the data. As observed during our our Data Exploration, some Reference Texts contain the keyword "*from*" followed by the BuPa Name. We extract everything that follows this keyword and perform Fuzzy Matching on the string segment against all possible BuPas. This approach contributes to runtime reduction since exempting from using a specific Keyword Extractor and the keyword list is essentially reduced to just one item. Additionally, we leverage country-specific currency information, excluding universal currencies such as the dollar, to narrow down the potential BuPa list for fuzzy matching. Thus reducing the search space significantly.

Another major optimization for runtime performance involves concatenating the BuPa name and additional name. This approach eliminates the need to scan the large BuPa list twice. It not only reduces the runtime by a factor of 18 on average, but also improves matching quality by filtering out noise results such as "ins".

## 3.6 Modelling Approaches for Machine Learning Mapper

So far, we have mainly relied on the Text Reference column for possible Direct Matching to identify BuPas. This means that the general accuracy depends highly on the quality of the Reference Text (i.e whether there was a reasonable identifier included). We want to further improve performance and tested another layer where we treat the problem as a supervised learning task.

**Supervised Learning:** Supervised learning means that a model is trained on a *labeled* dataset where each training example is paired with its output label. In our case one payment is one training example and the respective BuPas are the desired output labels. The goal of supervised learning is to predict the output label of *new, unseen* data based on this training data.

This implies directly that any supervised learning model can only ever predict the BuPas that were already *present in the training set.* Our historic training dataset covers around 10% of the total number of existing BuPas. This problem could theoretically be overcome if the train dataset is increased to take into account *every* company code instead of only the selected ones we received and increasing the time frame. A reduction of the BuPa List is not possible as it would not meet the requirements of accounting.

### 3.6.1 Random Forest and Gradient Boost

**Selecting a suitable model:** In terms of Machine Learning, our goal can be viewed as a classification task. However, the problem at hand is a classification to as many as over 7,000 classes in the unfiltered case (i.e. all historical and technically even future BuPas of Munich Re). This means that all binary classification models (e.g. Support Vector Machine) are not appropriate to pursue our goal. We therefore decided to test a Random Forest Classifier as well as Gradient Boosting in a first try.

We prepared the data as described in the previous section. However, the loss of information using Text Vectorization to transform the text data seems very high for these models as we achieved a low accuracy. The Reference Text is the most promising feature of the entire dataset as it contains the most exact and detailed information to identify the BuPa and these models did not seem to be able to leverage the information.

To overcome the high-dimensionality of the data caused by Text Vectorization, we tried to use *Dimensionality Reduction* techniques, such as *Principal Component Analysis* (PCA) to reduce the number of features. This idea is also used in *Transformers*, the state-of-the-art models for Natural Language Processing. Input text is vectorized and then an embedding into lower-dimensional space is searched for. However, employing PCA did not significantly improve performance of these two models.

To use the available information in the Reference Text more efficiently, we turned to models more suited for texts, in particular RNNs.

### 3.6.2 Imbalanced Dataset

In our very first approach to train a Machine Learning model on this problem, we experienced that one challenge will be the underlying imbalanced training dataset. We have already discussed that only a fraction of the possible BuPas is in fact present in the training set. However, the number of payments per BuPa also significantly fluctuates. In our Data Analysis we have seen that some BuPas have a lot of transactions while others may only be present with one single transaction. It is infeasible for any supervised learning model to be trained on only one individual example and learn a meaningful representation. The classes are consequently in no way balanced in our dataset. We are dealing with a multiclass imbalanced classification task, which will not perform optimally [3].

### 3.6.3 Recurrent Neural Networks

To make more out of the underlying data, we employed Recurrent Neural Networks (RNNs) next. RNNs are a class of neural networks that are particularly well-suited for processing sequential data. Unlike traditional feedforward neural networks, RNNs have connections that form directed cycles, allowing them to maintain a "memory" of previous inputs in the sequence. This makes them powerful for tasks involving *natural language processing* (Text Data), time series data, and any scenario where context or sequence order is important.

**Recurrent Neural Network Structure:** A RNN consists of input nodes ($x_0, x_1, x_2$), hidden states ($h_0, h_1, h_2$), and output nodes ($y_0, y_1, y_2$).

The arrows in the diagram indicate the flow direction of data through the network:

- From the input nodes to the hidden states (upward arrows).
- From one hidden state to the next (rightward arrows), allowing the network to maintain context over time.
- From the hidden states to the output nodes (upward arrows).
- Loops on the hidden states (circular arrows) indicate the recurrent nature, where the hidden state is used in subsequent steps.

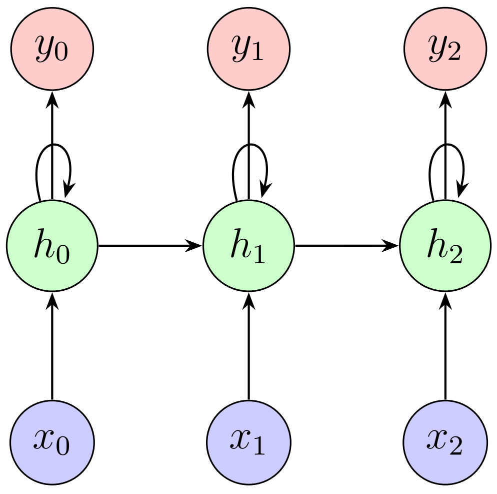


Figure 3: A SimpleRNN representation

Within this framework, we experimented with three different architectures:

1. **SimpleRNN:** Basic recurrent neural network that processes sequential data by maintaining a hidden state through time. It lacks advanced mechanisms like gates, making it simpler but less effective at capturing long-term dependencies.

2. **Long Short-Term Memory (LSTM):** RNN architecture with memory cells and gates (forget, input, and output gates) designed to capture and remember long-term dependencies in sequential data, commonly used in tasks like language modeling and speech recognition [6].

3. **Gated Recurrent Unit (GRU):** Simplified RNN variant that merges the cell state and hidden state into a single vector, with update and reset gates controlling the flow of information [6]. It balances performance and computational efficiency, often preferred for tasks where both speed and effectiveness in capturing dependencies are crucial.

Various parameters were adjusted during the experimentation process, including:

- **Epochs:** The number of complete passes through the training dataset. We tested up to 100 epochs.

- **Activation Functions:** Functions that determine the output of a neural network node. Different activation functions such as sigmoid and softmax were used.

- **Embedding Layers:** Used to transform categorical data into continuous vector spaces. We experimented with up to 3 layers.

- **Neurons per Layer:** We used up to 128 neurons per layer.

**Regularizing Recurrent Neural Networks:** Regularization refers to controlling the capacity of the neural network by adding or removing information to prevent overfitting. For better training of a RNN, a portion of available data is considered as a validation dataset. The validation set is used to watch the training procedure and prevent the network from underfitting and overfitting. Overfitting refers to the gap between the training loss and the validation loss which increases after a number of training epochs as the training loss decreases.[**RNNs**]

### 3.6.4 Other Tested Models

We further conducted extensive testing on various Neural Network Models to evaluate their performance on our dataset. Specifically, we examined Multiple Layer Perceptrons (MLP), Deep Neural Networks (DNN), and Generalized Additive Models (GAMs). However, these complex models yielded disappointing results, with accuracy rates falling below 35%. Given that these sophisticated models failed to provide an accuracy boost, we shifted our focus towards reducing model complexity.

## 3.7 Decision Tree

After experiencing that more intricate models did not achieve the desired results, we turned again to a very simple model: the Decision Tree.

Decision Trees are one of the most interpretable Machine Learning methods. The resulting tree can be easily visualized, and the decision rules can be readily extracted, making it highly transparent and explainable. Decision Trees typically create splitting criteria based on the highest information gain. Entropy measures the impurity of the dataset, i.e., it quantifies how the different classes (labels) are distributed in the present dataset. Information gain is defined as the difference between the entropy of the dataset before splitting and the entropy after splitting. This measure helps the tree determine the most effective way to separate the data at each node, maximizing the reduction in uncertainty about the target variable. Each node in the tree represents a test of a specific criterion. Outgoing edges usually represent a *binary split*, dividing the dataset into two parts: one where the criterion is fulfilled and another where it is not (see equation 1).

$$Entropy(X) = -\sum_{c \in C} p(c) log_2 p(c) \tag{1}$$

where C is the set of classes the target feature can take (e.g. all BuPas present in the training set) [5]. If the dataset would only contain samples of one class then $p(c) = 1$ and therefore $log_2 p(c) = 0$, which means the entropy of the dataset is equal to 0. The entropy will be maximal (i.e. equal to 1) if all the classes are distributed equally among the dataset. For a binary target, this would mean the dataset contains 50% samples labelled as 'yes' and the other half labelled as 'no'.

However, Decision Trees can easily *overfit*, they tend to memorize the training data and perform bad on unseen data. This can be overcome by *pruning* the tree and choosing a maximal depth for the tree. These are hyperparameters that need to be tuned. To find optimal ones, we employed an *extensive* grid search. In the end, this approach turns out to give us the best results.

# 4 Results

## 4.1 Benchmarking on Direct Mapping

**One-to-One ID** Given numeric search is very fast on very modern hardware, we were able to benchmark the One-to-One ID matching on the full list of historical data. The results show `4.49%` of the entries can be matched directly via BuPa ID and Treaty Number, which reinforces the importance of our composite design.

**Keyword Indian Companies** Some special cases enabled us to leverage specific business knowledge. All Indian Companies use a special keyword in the Reference Texts. This makes it possible to narrow down the BuPa List.

**Benchmark Creation** To create a representative benchmark dataset, we randomly selected 5% of the singleton cases (i.e., BuPa IDs with only one associated transaction) and combined them with 5% of the multi-transaction cases, which were selected using stratified sampling. This approach ensured that the test set captured both the common and the more challenging examples to evaluate the fuzzy matching design performance, especially considering same BuPa tends to provide same or similar reference text.

**Fuzzy matching via keywords** Fuzzy Matching Names and previous extraction by Keyword Extractors needs long computation time. The very first computation time took at least 4 minutes per entry. However, as illustrated in Figure 5, we implemented several optimization techniques (see section 3.5.4) that significantly reduced this computation time. To evaluate the performance of fuzzy matching and the effectiveness of these enhancements, we benchmarked the fuzzy matching results using the created benchmark dataset.

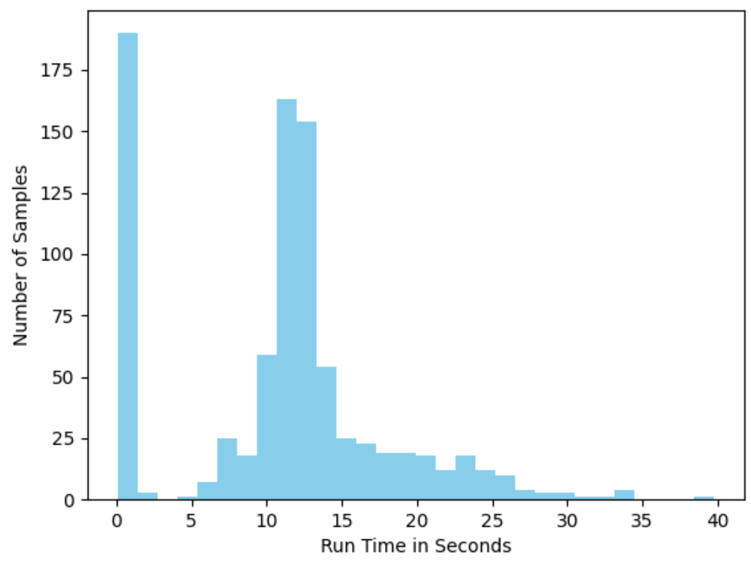


Figure 4: Distribution of Run Time for Fuzzy Matching

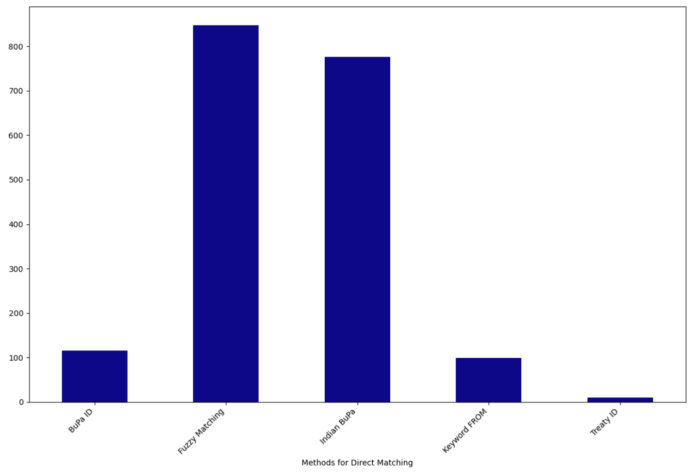


Figure 5: Methods used to for possible Matches during Direct Matching in the Benchmark

Overall, we could achieve an accuracy of 54% on the Benchmark Sampled Data via Direct Matching leveraging these methods. Figure 6 shows that the most cases were matched via Fuzzy Matching.

## 4.2 RNN Baseline Model

The feature used for the RNN was the pre-processed text and the target was the BuPa ID. Despite all the efforts, all architectures exhibited evident overfitting. To mitigate this, we employed early stopping, a technique that stops training when the model's performance on a validation set starts to degrade thereby preventing overfitting.

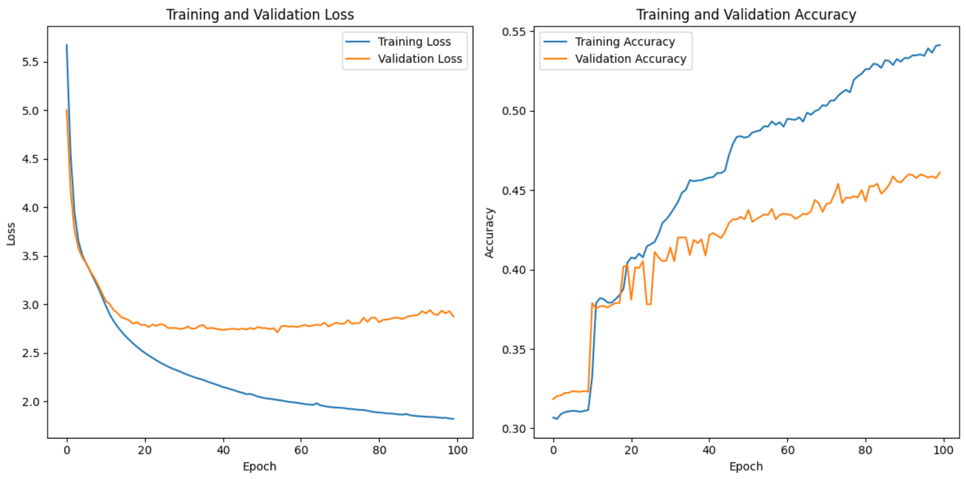


Figure 6: Overfitting in training neural networks.

After training the model on 10 different versions of the dataset obtained by cross validation, we achieved an average validation accuracy of 43%.

## 4.3 Machine Learning Mapper

In the end, a Decision Tree gave us the best results of all Machine Learning Methods tested. For the Decision Tree, we used seven different features (see section 3.2). To train and evaluate the model as objective as possible given the imbalanced datasets, we employed a cross-validation approach using 10 subfolds. The first baseline Tree model with common max-depth of 12 gave us an accuracy of around 50%. After employing an extensive grid search to further fine-tune the hyperparameters, combining other optimization technique, we could boost the accuracy to 66%.

This higher accuracy was also possible since we *grouped BuPa Names* to address the issue of imbalanced data. We simply matched identical BuPa Names that were mapped to several distinct BuPa IDs. It is worth noting that due to Munich Re being a multinational company, the BuPas could use the same name but reside in different countries, which enabled this technique. The resulting groups were not large, usually consisting of two BuPa IDs. The Decision Tree would then return the list of IDs which is counted as correct if the true BuPa is present in the list. We could see that this significantly reduced the number of possible classes and therefore the performance of our Machine Learning model.

# 5 Discussions

## 5.1 Imbalanced Data Issue

As we discussed before here, one problem hindering performance is having imbalanced classes in our dataset. Of course, there exist different techniques to work-around this problem. However, we want to discuss in this section why they were not feasible to be used in our particular problem.

**Data Sampling:** One standard approach is simply resampling the data of the underrepresented classes. However, as shown here before, for almost 20% of the BuPas present in the dataset, there is only a single transaction. Duplicating these would mean any model would only identify this single copied transaction as the corresponding BuPa.

**Creating Synthetic Data:** To overcome the problem of imbalanced classes, it is theoretically possible to create synthetic data. When doing for example image classification, one can simply crop or rotate images to increase the training set size. However, in our case it is not as trivial to generate new data as specific accounting knowledge needs to be leveraged. We leave that for Future Directions.

**Grouping the BuPas:** To simplify the problem further and to increase performance, we started to group some of the BuPas. We used a very simple clustering approach, which is to simply match *identical* BuPa Names, that appear to have *multiple* BuPa IDs. Most of these groups will contain only two BuPa IDs. The Model will return the grouped IDs out of which a Technical Accountant then is able to pick. This partially overcomes the challenge of having too many classes to classify to and too little training samples of some classes.

It is possible to further broaden this technique by additionally Fuzzy Matching BuPa Names instead of only grouping exact matches. This would increase the cluster size (and therefore the number of training samples) and simultaneously reduce the number of possible classes to predict to. For example, this would cluster companies and its subsidiaries where the names are `[Company Name][Country Name]`.

## 5.2 Solution Merits

**Flexibility of Modularized Design** The modularized design of this solution offers significant flexibility and efficiency advantages. By separating the process into a Direct Mapping component and a Machine Learning Mapper component, the system can quickly handle straightforward matches while reserving more complex cases for advanced analysis. Although the Machine Learning component is currently trained on data from 12 companies, the Direct Mapper's capability to process English-based payment histories provides a degree of universality. This modular structure also enables easy expansion or modification of individual components as needed, thereby enhancing the solution's overall adaptability and longevity.

**Fast Training** Given the intrinsic structure of the Decision Tree (DT) model, training is remarkably fast, typically taking less than 4 minutes. This rapid training time makes continuous retraining highly feasible and cost-effective, allowing the system to quickly adapt to new patterns in incoming payments and changes in business partners. As a result, the accuracy of the system remains consistently high over time.

## 5.3 Limitation and Continuous Training

We again want to emphasize that also a Decision Tree is a Supervised Learning Model. It can never predict an unseen BuPa that was not present in the training data. Therefore a continuous training of the model is necessary. With our solution, an unseen BuPa could only be identified via the Direct Matching if the Reference Text is meaningful.

**Adding New BuPas:** The BuPa List of Munich Re obviously is not a *fixed* list. Munich Re will make contracts with new BuPas in the future. This means that our model would need to be retrained periodically whenever new transactions are available. Consequently, this should improve future performance, as additional training data will be available.

One could also consider adding new BuPas to one new additional Cluster which could then be returned if no match in the old training data is found. However, this becomes infeasible with an increasing number of new BuPas.

**Mergers of BuPas:** Apart from new BuPas, it can also happen that two (or more) companies will merge, e.g. one parent company and its subsidiaries. The BuPa List will need to be updated again.

## 6 Future Directions

To enhance our model's performance and address the issue of imbalanced datasets, we grouped BuPa IDs by Matching their underlying Names as explained above. It is possible to further improve performance by using more sophisticated grouping techniques such as fuzzy matching BuPa name to reduce number of single transactions for training. However, this may require significant processing time and memory due to the extensive list of business partners with both Name and Additional Name attributes. Another potential method is to explore the creation of *Synthetic Data* for the single transaction, as we discussed in section 5.1. This is possible if one leverages specific accounting knowledge.

As we have seen in some cases, our model will return a list of possible BuPas instead of one single match. This gives the opportunity for Technical Accountants to select the correct BuPa. One could think about returning this feedback to the model again and retrain it based on this decision. This will turn the task in some kind of Reinforcement Learning.

During the project we have encountered several challenges. Inconsistencies in the underlying data made it hard for us to find one single convincing model. We therefore explored several modeling approaches and tried to exploit each of its advantages. In the end, we have achieved a decent accuracy rate nevertheless.

## References

[1] Rishin Haldar and Debajyoti Mukhopadhyay. “Levenshtein distance technique in dictionary lookup methods: An improved approach”. In: *arXiv preprint arXiv:1101.1232* (2011).

[2] KeyBERT. *https://maartengr.github.io/KeyBERT/index.html*.

[3] Bartosz Krawczyk. “Learning from imbalanced data: open challenges and future directions”. In: *Progress in Artificial Intelligence* 5.4 (2016), pp. 221–232.

[4] Munich Re. “Annual Report”. In: *www.munichre.com/en/company/investors/reports-and-presentations/annual-report.html* (2023).

[5] Lior Rokach and Oded Maimon. “Decision trees”. In: *Data mining and knowledge discovery handbook* (2005), pp. 165–192.

[6] Hojjat Salehinejad et al. “Recent advances in recurrent neural networks”. In: *arXiv preprint arXiv:1801.01078* (2017).

[7] SpaCy. *https://spacy.io/*.

[8] Rüdiger Wirth and Jochen Hipp. “CRISP-DM: Towards a standard process model for data mining”. In: *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. Vol. 1. Manchester. 2000, pp. 29–39.

[9] YAKE. *https://liaad.github.io/yake/*.