AI

MDPI

Article

A Bag-of-Words Approach for Information Extraction from Electricity Invoices

Javier Sánchez *,† and Giovanny A. Cuervo-Londoño †

Centro de Tecnologías de la Imagen (CTIM), Instituto Universitario de Cibernética, Empresas y Sociedad (IUCES), 3507 Las Palmas de Gran Canaria, Spain; giovanny.cuervo101@alu.ulpgc.es

* Correspondence: jsanchez@ulpgc.es; Tel.: +34-928-458-710

† Current address: Department of Computer Science, University of Las Palmas de Gran Canaria, 35017 Las Palmas de Gran Canaria, Spain.

Abstract: In the context of digitization and automation, extracting relevant information from business documents remains a significant challenge. It is typical to rely on machine-learning techniques to automate the process, reduce manual labor, and minimize errors. This work introduces a new model for extracting key values from electricity invoices, including customer data, bill breakdown, electricity consumption, or marketer data. We evaluate several machine learning techniques, such as Naive Bayes, Logistic Regression, Random Forests, or Support Vector Machines. Our approach relies on a bag-of-words strategy and custom-designed features tailored for electricity data. We validate our method on the IDSEM dataset, which includes 75,000 electricity invoices with eighty-six fields. The model converts PDF invoices into text and processes each word separately using a context of eleven words. The results of our experiments indicate that Support Vector Machines and Random Forests perform exceptionally well in capturing numerous values with high precision. The study also explores the advantages of our custom features and evaluates the performance of unseen documents. The precision obtained with Support Vector Machines is 91.86% on average, peaking at 98.47% for one document template. These results demonstrate the effectiveness of our method in accurately extracting key values from invoices.

Keywords: electricity invoice; information extraction; semi-structured document; machine learning; support vector machine

Citation: Sánchez, J.; Cuervo-Londoño, G.A. A Bag-of-Words Approach for Information Extraction from Electricity Invoices. *AI* **2024**, *5*, 1837–1857. https://doi.org/10.3390/ai5040091

Academic Editor: Arslan Munir

Received: 25 July 2024
Revised: 20 September 2024
Accepted: 30 September 2024
Published: 8 October 2024

1. Introduction

Extracting information from semi-structured documents is essential in many business processes, as organizations often need to automatically retrieve data to feed their Enterprise Resource Planning (ERP) systems. This is typically conducted manually, being a time-consuming task prone to errors [1]. This problem has been a topic of interest in machine learning and data science for many years [2]. The complexity of invoice formats and the variability in data presentation make this a challenging task.

Electricity invoices, in particular, are special business documents that contain various information. These documents are particularly intricate due to the complexity of electricity markets. These markets usually serve millions of customers, involve multiple stakeholders, such as marketers and distributors, and are subject to specific national regulations. As a result, electricity bills contain diverse and abundant data. Therefore, the main difficulty arises from documents with varying contents and layouts.

Several commercial applications, such as DocParser [3], ITESOFT [4], or ABBYY [5], may facilitate the task of information extraction. They allow invoice information to be retrieved automatically, although they typically require the users to configure template bills manually. This information is incorporated into databases, which later permit the identification and processing of similar invoices. Nevertheless, it does not usually scale well

to new types of documents. Furthermore, current machine learning techniques are often designed to recover just a few generic fields, such as invoice numbers or total amounts, so even if commercial solutions exist, this problem is not solved yet.

There are several types of methods for extracting information from invoices. Probably the most traditionally used involve user-defined templates [1] and the use of predefined-rules [4]. However, these techniques typically need to define templates by hand and do not adapt well to new types of documents. On the other hand, some methods are based on custom-designed features and machine-learning techniques [6,7]. They usually adapt well to a given set of templates and do not need a large dataset to train on.

Another type of method relies on classification through Named Entity Recognition (NER) [8], usually combined with neural networks [9,10]. Recent advancements in deep learning have led to the use of more complex models, such as Trie [11] and LayoutLM [12], which integrate visual and textual information, addressing the challenge of handling both text and images in a unified framework. This can improve accuracy and adaptability because of the visual structure of invoices. However, a drawback of these methods is that they need to be trained on large datasets and it is often complex to adapt the samples to the format required by these neural networks. Most methods are designed for processing simple invoices and extracting a few fields, such as the invoice number or the total amount. Adapting them for more complex documents and incorporating additional labels poses a considerable challenge.

Our work is based on a database of electricity invoices, called IDSEM [13], containing 75,000 invoices. This dataset was created randomly using information from the Spanish electricity market, based on regulations and statistical data. This dataset utilizes eighty-six distinct labels, significantly larger than similar datasets.

The main goal of our paper is to explore and compare different algorithms for extracting information from electricity invoices, which are challenging to process due to their complex structure and varied formats. We propose a method for extracting information from invoices using a bag-of-words approach and standard machine-learning techniques. The bag-of-words approach relies on Term Frequency and Inverse Document Frequency (TF-IDF) features [14]. By analyzing the frequency of words, we can determine their importance within the context of the invoice. Then, we characterize each word based on its format: whether it is numeric or alphanumeric, has a predefined format (e.g., money or postal codes), is a measurement unit (e.g., kW or €/kW), includes capital letters, or contains punctuation marks.

This study evaluates the performance of various classification techniques, including Naive Bayes, Logistic Regression, Support Vector Machine (SVM), and Random Forests. Initially, invoices are converted from PDF to raw text. Then, we create lists of eleven-word sentences using a sliding window, with consecutive sentences differing in the first and last words. These sentences are passed through a pipeline composed of TF-IDF, our custom-designed features, and a classification technique. The assigned label corresponds to the central word of each sentence; therefore, we obtain a classification for every word in the document.

The experimental results provide a comparative analysis of the performance of our model using different machine-learning techniques. They show that Support Vector Machines achieve the highest precision (91.86%), closely followed by Random Forests (91.58%). The study also examines the performance of methods on documents similar to the training set and on completely new documents. The results indicate that Random Forests show better adaptability to new documents.

Most labels are classified with high precision, with over 41% of labels having precision rates exceeding 99% and another 31.3% with precision rates above 90%. The experiments also investigate the benefits of using the bag-of-words approach and our custom features. The successive integration of our custom features proves crucial for enhancing the performance of methods. Combining both types of features provides the best results for the eleven-word sentences.

The main contributions of this work can be summarized as follows: (1) we propose a new model based on TF-IDF and custom-designed features for extracting a large number of key values from invoices; (2) we transform a complex dataset into a text-based format that is suitable for classifying the labels of these documents; (3) our proposed custom-designed features provide a high classification rate for the contents of electricity invoices; (4) we analyze the performance of different machine-learning techniques based on this model; (5) we realize a thorough analysis of the benefits of each type of feature and a detailed evaluation assessment of the methods regarding each label and template. We also discuss the advantages and drawbacks of our approach and provide some guidance to improve it.

This model offers several advantages over existing methods. Primarily, it is designed to handle documents with a large number of fields, which is not typical in the literature where most methods focus on simpler receipts and invoices with few labels, typically less than ten. Another advantage is the specificity of electricity invoices. Unlike general information extraction methods, our approach is specifically tailored to handle the content of electricity invoices. These documents contain a wide variety of data from different sources, making them more complex than other invoices. The adaptability of the model to new document types is another advantage, as it uses a general pipeline based on textual information, allowing for easy adaptation without significant changes to the process.

Section 2 describes similar methods and summarizes the state-of-the-art. Section 3 describes the IDSEM dataset and the conversion of PDF documents into the training data. Section 4 explains the details of the TF-IDF and our custom features. Section 5 describes the classification methods and Section 6 explains the metrics employed and the experimental design. The results in Section 7 analyze the performance of the methods using standard metrics, such as precision, recall, and F1-scores, paying special attention to the influence of the features and the precision regarding new documents. We discuss the results of the method and some future works in Section 8.

2. Related Work

Information extraction is an active research field with applications in various domains such as historical document analysis, web content mining, or business document processing. Over the past two decades, researchers have proposed many methods to tackle this problem, with more interest in machine learning techniques nowadays.

Multiple datasets and benchmarks have played an important role in advancing this field. For instance, the SROIE dataset [2], one of the earlier receipts datasets, contains 1000 scanned receipt images and is used for three main tasks: text localization, OCR, and key information extraction. Another receipts dataset is CORD [15], which contains thousands of Indonesian receipts, including images and text annotations for OCR, and multi-level semantic labels for parsing. These datasets contributed to the development of many methods but they contain simple receipts with a few annotated fields.

DocILE [16] is another dataset with over 6700 annotated business documents, manually labeled and synthetically generated. It provides annotations for fifty-five classes, covering various document layouts. More recently, the benchmark proposed in [17] provided two new datasets, HUST-CELL and Baidu-FEST, that aim to evaluate the end-to-end performance of complex entity linking and labeling, containing various types of documents, such as receipts, certificates, licenses of several industries, and other types of business documents. The IDSEM dataset [13] that we use in this work contains 75,000 synthetically generated invoices related to electricity consumption in Spanish households. It has a large number and variety of labels. Each invoice includes data about customers, contracts, electricity consumption, and billing that are challenging to classify.

The first methods for extracting information from invoices involved user-defined templates and the use of rules. This was the case of SmartFIX [1], which was among the earliest systems in production for processing business documents, ranging from fixed-format forms to unstructured letters. The work proposed in [18] presented a method for automatically indexing scanned documents, reducing the need for manual configuration.

The authors leveraged document templates stored in a full-text search index to identify index positions that were successfully extracted in the past. The authors of [4,19] proposed an incremental learning approach based on structural templates. The document was recognized by an OCR and then classified according to its layout. The fields were extracted using structural templates learned from the layout. The lack of information detected by the users was used to improve the templates in an incremental step. These methods require the users to create new templates manually and check the final results.

Information extraction can also be tackled through NER [20], a typical step in any NLP pipeline. In this case, words in the text are tagged with specific labels. Some typical examples [8] relied on Long Short-term Memory (LSTM) neural networks [9,21]. The method proposed in [22] compared two deep learning approaches, the first based on NER using a context of words around each label, and the second based on a set of features. This second method provided similar results with less training data. In the preliminary work presented in [10], the authors obtained high accuracy for a reduced set of labels using the IDSEM dataset. They also used an NER strategy. The implementation was realized through a neural network based on the Transformer [23] architecture. These strategies typically work correctly for a few labels and the vocabulary needs to be similar for all documents.

Another group of methods relies on custom-designed features and shallow machine-learning techniques. For example, Intellix [6] was based on TF-IDF features and K-Nearest-Neighbors (kNN). In our work, we use a similar approach, with a larger variety of features, and analyze the behavior of more classification techniques. Another example was the method presented in [7], which converted documents to JPEG images and extracted word tokens through an OCR system with their positions. This method relied on a gradient-boosting decision tree model for the classification step. In the experiments, we use Random Forest, another ensemble method based on decision trees. We choose these techniques as they give us the flexibility to choose the most convenient features and adapt well to the problem that we are facing.

More recently, several methods based on deep-learning techniques have been proposed. Various approaches [24,25] combined convolutional neural networks (CNN) with a spatial representation of the document using a sparse 2D grid of characters. This allows for understanding the semantic meaning and the spatial distribution of texts. Other methods are based on LSTM networks, like CloudScan [26], or a combination of CNNs and LSTMs, like in [11,27,28]. The latter presented a multi-modal end-to-end information extraction framework based on visual, textual, and layout features. However, this requires the dataset to be annotated with bounding boxes around the key values, and the IDSEM dataset does not provide this information.

The Transformer [23] has also been employed in several works, like in XLNet [29] that integrated ideas from the Transformer-XL [30] architecture, which enabled learning for longer-length contexts. DocParser [3] was an end-to-end solution based on a Swin Transformer [31] comprising a visual encoder followed by a textual encoder. BERT [32] was also adapted to information extraction in BERTgrid [33]. The most recent methods [12,34,35] relied on multi-modal pre-training models integrating text and images, or methods that establish geometric relations during pre-training, like in [36].

These methods are complex and rely on visual annotations to train the models. However, the IDSEM dataset contains textual annotations and is difficult to adapt for training with these networks. In our model, invoices are converted to text format and the features are designed at the sentence level, being a simple and efficient solution.

3. Electricity Invoice Dataset

In this section, we explore the IDSEM dataset, providing some statistics about the labels. Then, we explain the data transformation from PDF invoices into the training sentences.

3.1. IDSEM Dataset

The IDSEM dataset [13] contains 75,000 electricity invoices in PDF format, with 30,000 in the training set and 45,000 in the test set, for which the ground truth values are not publicly available. Each invoice is defined with eighty-six labels related to customer data, the electricity company, the breakdown of the bill, the electricity consumption, etc. This dataset is available at Figshare [37].

The labels are defined with a code starting with a letter (`A`–`N`) and a number for each label in the group. Table 1 summarizes the groups of labels of the dataset and their meaning is explained in [13].

Table 1. Groups of labels of the IDSEM dataset. The first column shows the letter code that identifies each group, the second column summarizes the contents of the labels in the group, and the third column enumerates the total number of labels.

ID	Description	Number of Labels
`A`	Customer who receives the invoice letter	6
`B`	Customer data as stated in the contract	6
`C`	Marketer data	14
`D`	Distributor data	5
`E`	Contract information	9
`F`	General information about the invoice	8
`G`	Customer financial information	10
`I`	Energy consumption information	3
`J`	Summary of invoice breakdown	5
`K`	Detailed invoice breakdown	10
`M`	Other billing items, like equipment rental	2
`N`	Taxes	8

The dataset includes a training directory with six sub-directories for different template documents. Each directory has 5000 invoices in PDF format and their corresponding JSON files with the labels and their true values. The test directory contains nine sub-directories with nine different templates, three of them completely different from the training set. These allow us to study the behavior of the method to unseen data.

Figure 1 depicts two pages of an invoice from the dataset. Blue boxes relate the groups of labels and their position in the document.

Table 2 shows the frequency of occurrence of several labels in the documents of the training set. For example, the label `C1`, corresponding to the marketer's name appears 564,700 times. Therefore, this information appears multiple times in each invoice. On the other hand, label `M3`, corresponding to the equipment rental price, appears 5000 times. In this case, the label appears only in one template of the training set. Additional statistics for all the labels can be found in Appendix A.

Table 2. Number of times the labels appear in the training set documents.

Labels	#Occurrences	Labels	#Occurrences	Labels	#Occurrences
`C1`	564,700	`C9`	143,600	`CC`	105,600
`N4`, `N6`	60,000	`A4`, `G4`	35,000	`F2`, `KD`, `K6`	20,000
`E5`, `K9`, `KA`	15,000	`K2`, `K3`, `K5`	10,000	`G6`, `K4`, `M3`	5000

in the header stands for *Number of*.

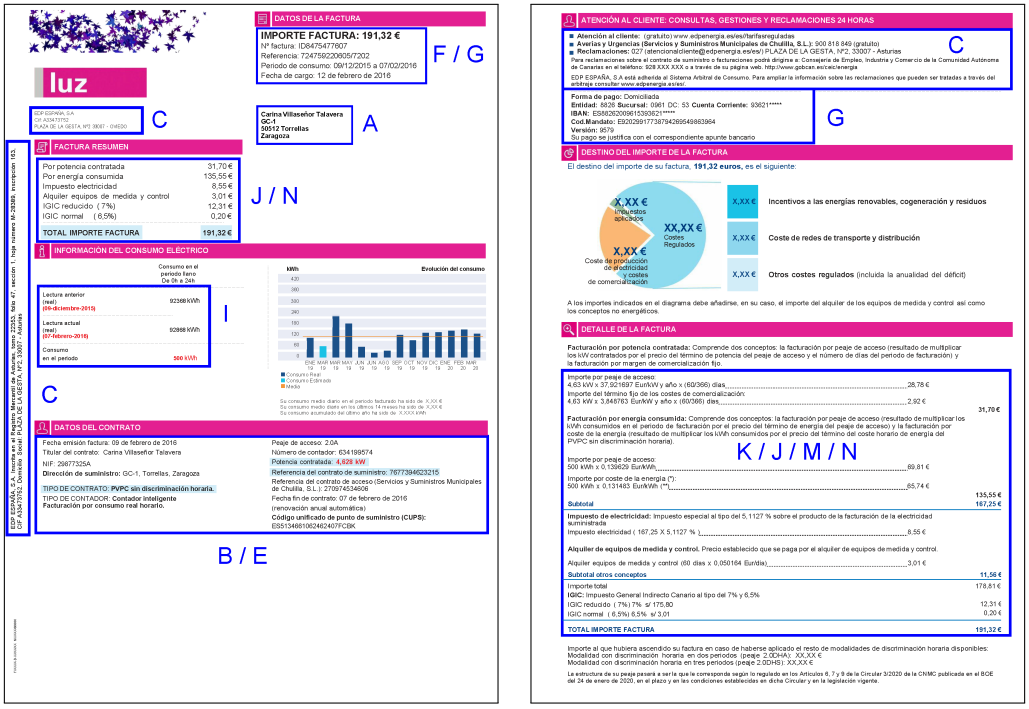

Figure 1. Example of an electricity invoice. The contents of the document are organized by customer data (`A`), marketer information (`C`), the breakdown (`B`, `E`), a summary of the bill amounts (`J`, `N`), contract information (`F`, `G`), the electricity consumption (`I`), breakdown of the invoice (`K`), and other billing concepts (`M`). Source : Sánchez et al. [13].

3.2. Data Conversion

Similar to other approaches [26], our method converts the input data into text using a PDF-to-text converter. The end-of-line character is replaced by a special code, `#eol`, to ensure the invoice is treated as a continuous list of words. We generate eleven-word sentences, using a sliding window of one word between consecutive sentences. The label is associated with the central word of the sentence. Our method is complete as it predicts a label for every word in the document.

Using a sliding window approach enables the classification of individual words within a document. While this method can be time-consuming, its efficiency can be significantly improved through parallel processing. By parallelizing the processing of each sentence, the overall execution time can be reduced to approximately that of a single sentence. On the other hand, we chose eleven words per sentence because this provided the best precision, as shown in the experiments.

We utilize a special label, `OO`, standing for *others*, to classify sentences for which the central word does not match any label in the dataset. This allows the model to deal with out-of-class words efficiently. The `OO` label is introduced during the conversion of PDF documents to our internal representation of sentences. This common strategy is used frequently to integrate this type of token in the classification process, without modifying the main pipeline, making it simpler and more efficient. This robust approach allows the model to generalize to new unclassified words. We generate a training file for each template directory, containing fifty invoices, resulting in six training files with 300 documents.

These files consist of tuples with two fields: one with an eleven-word sentence and another with the label corresponding to the central word. Therefore, the number of sentences coincides with the total number of words. For this process, we use the annotated files of the training set, which include the code of the labels around each key value. We parse the file contents, remove the annotations from the text, and generate the tuples. A fragment of a training file in our format is shown in Table 3.

Table 3. Fragment of a training file. We create a file with a list of tuples obtained from an annotated PDF file. Each tuple contains a sentence of eleven words from the bills and the label corresponding to the central word (in bold letters). Sentences are created with a one-word sliding window. The label `OO` is used for words that do not correspond to any label in the dataset.

Sentence	Label
FACTURA #eol TOTAL FACTURA : **90.31** #eol Factura Nº : XT38563865H	`J5`
#eol TOTAL FACTURA : 90.31 **#eol** Factura Nº : XT38563865H #eol	`OO`
TOTAL FACTURA : 90.31 #eol **Factura** Nº : XT38563865H #eol K	`OO`
FACTURA : 90.31 #eol Factura **Nº** : XT38563865H #eol K #eol	`OO`
: 90.31 #eol Factura Nº **:** XT38563865H #eol K #eol Ref	`OO`
90.31 #eol Factura Nº : **XT38563865H** #eol K #eol Ref :	`F1`
#eol Factura Nº : XT38563865H **#eol** K #eol Ref : 72028535275J	`OO`
Factura Nº : XT38563865H #eol **K** #eol Ref : 72028535275J Periodo	`OO`
Nº : XT38563865H #eol K **#eol** Ref : 72028535275J Periodo de	`OO`
: XT38563865H #eol K #eol **Ref** : 72028535275J Periodo de consumo	`OO`
XT38563865H #eol K #eol Ref **:** 72028535275J Periodo de consumo :	`OO`
#eol K #eol Ref : **72028535275J** Periodo de consumo : 3/5/2017	`F2`
K #eol Ref : 72028535275J **Periodo** de consumo : 3/5/2017 a	`OO`
#eol Ref : 72028535275J Periodo **de** consumo : 3/5/2017 a 5/07/2017	`OO`
Ref : 72028535275J Periodo de **consumo** : 3/5/2017 a 5/07/2017 Fecha	`OO`
: 72028535275J Periodo de consumo **:** 3/5/2017 a 5/07/2017 Fecha de	`OO`
72028535275J Periodo de consumo : **3/5/2017** a 5/07/2017 Fecha de cargo	`F4s`
Periodo de consumo : 3/5/2017 **a** 5/07/2017 Fecha de cargo :	`OO`
de consumo : 3/5/2017 a **5/07/2017** Fecha de cargo : 7	`F5s`
consumo : 3/5/2017 a 5/07/2017 **Fecha** de cargo : 7 de	`OO`
: 3/5/2017 a 5/07/2017 Fecha **de** cargo : 7 de Julio	`OO`
3/5/2017 a 5/07/2017 Fecha de **cargo** : 7 de Julio de	`OO`
a 5/07/2017 Fecha de cargo **:** 7 de Julio de 2017	`OO`
5/07/2017 Fecha de cargo : **7** de Julio de 2017 #eol	`G3`
Fecha de cargo : 7 **de** Julio de 2017 #eol #eol	`G3`
de cargo : 7 de **Julio** de 2017 #eol #eol EDP	`G3`
cargo : 7 de Julio **de** 2017 #eol #eol EDP ESPAÑA	`G3`
: 7 de Julio de **2017** #eol #eol EDP ESPAÑA ,	`G3`
7 de Julio de 2017 **#eol** #eol EDP ESPAÑA , S.A	`OO`

In this example, we observe that various sentences refer to a single label value, such as the total amount (`J5`), the invoice number (`F1`), or various dates (`F4s` and `F5s`), and several sentences refer to the same label, such as the due date (`G3`). Label `OO` appears with high frequency since the central word of many sentences does not correspond to any label.

4. Definition of Word-Level Features

We employ a bag-of-words approach based on TF-IDF [14] features. These are often used in NLP and information retrieval to assess the importance of a term within a document relative to a corpus of documents. Term Frequency (TF) measures how often a term t appears in a specific document d and is calculated as the ratio of the number of times a term appears in a document to the total number of words in that document:

$$\mathrm{TF}(t,d) = \frac{\text{count of } t \text{ in } d}{\text{number of words in } d}. \quad (1)$$

Document Frequency (DF), on the other hand, assesses how common a term is across the entire corpus and is calculated as the number of documents that contain a term t, i.e., $\mathrm{DF}(t) =$ occurrence of t in documents.

Inverse Document Frequency (IDF) measures the relevance of a term and is the inverse ratio of the total number of documents to the document frequency of the term:

$$\mathrm{IDF}(t) = \log\left(\frac{N}{\mathrm{DF}(t)}\right), \tag{2}$$

where N is the total number of documents. TF-IDF is then measured as $\text{TF-IDF} = \mathrm{TF} \times \mathrm{IDF}$. Words with higher TF-IDF scores are considered more significant within a document.

Due to the limited vocabulary of electricity invoices, this method utilizes fewer tokens. We found that using single words and bi-grams improved classification precision in the experiments. Furthermore, the vocabulary obtained by the TF-IDF estimator using the training data contains many irrelevant words. Therefore, we define a dictionary of stop-words to improve the performance of this technique. This dictionary includes articles, prepositions, pronouns, and other frequent words in electricity invoices that are not relevant.

These features are insufficient to achieve high scores due to the limited number of words per sentence and the significant overlap between consecutive sentences. To address this issue, we designed a set of custom features that enable us to generate more distinctive data. One of these features is based on the word type, i.e., whether it contains alphabetic, numeric, or alphanumeric characters, or is a decimal number. Another feature is related to the format of the word, checking if its structure resembles money, email or web URLs, Spanish identity card numbers, postal codes, or date formats. We have also included features that provide information about different units, such as money (Euro symbol and other currency-related words), kilowatt (kW), kilowatt-hour (kWh), day and month periods, and the percentage symbol. Additionally, we have incorporated a feature to identify capital letters, which are often relevant in various contexts. Finally, we have defined another feature to identify punctuation symbols, such as colons, semicolons, or periods, as they typically convey relevant information, especially in tabular data.

These features are tailored to characterize the kind of words that we usually find in electricity invoices and are associated with the information we want to extract. These are specially designed for the Spanish locale, although they can be easily generalized for other regions. Table 4 summarizes these features.

Table 4. Custom features. The method relies on five custom features calculated for each word in the training sentences. These features model different aspects of the words that help characterize the information to be extracted. The first column defines the type of data and, the second column, the values to be characterized.

Feature Type	List of Analyzed Terms
Word type	Alphanumeric, Alphabetic, Numeric or Decimal number
Word format	Money, Email, Website, Identity card number, Postal code or Date formats
Units	€, €/day, €/kW, €/kWh, kW, kW/month, kWh, %
Capital letters	First, one, or all letters capitalized, Lowercase letters
Punctuation	Colons, Semicolons, Periods

Therefore, each word in the sentence produces a set of five custom features concatenated with the above TF-IDF features. In the experiments, we evaluate the benefits of using custom features only for the central word in the sentence or up to the eleven words in the sentence.

5. Classification Methods

In this work, we select the following machine-learning models for classification: Logistic Regression, Naive Bayes, Decision Trees, Random Forest, and SVM. These techniques receive a vector of features for each sentence and produce a label associated with the central word. Therefore, our method is a multi-class classification problem, where an input sequence produces a single label.

These are standard methods in machine learning and are typically used in many works. They cover a wide range of techniques with different characteristics. These methods can solve similar problems efficiently and are easy to interpret and implement.

Logistic Regression can be used for multi-class classification using a one-vs-all approach. It uses a sigmoid function that outputs a value between 0 and 1 to model the probability of an instance belonging to a positive class. For multiple classes, separate models are trained for each class. The instance is assigned to the class with the highest predicted probability. We also utilize a low regularization parameter to prevent overfitting.

Naive Bayes is a probabilistic algorithm that assumes feature independence. It calculates the probability that an instance belongs to a specific class. The steps involve computing prior probabilities, class conditional probabilities, and assigning the instance to the class with the highest posterior probability. We rely on the Multinomial Naive Bayes method typically used in text classification and a low regularization parameter.

Decision Trees are decision support tools used for both regression and classification tasks. They create a tree-like structure where each node represents a decision based on the input features. In multi-class classification, decision trees split nodes to predict class labels. The process involves calculating probabilities, branching, and assigning instances to the class with the highest posterior probability. The Gini criterion is used to split the data at the nodes.

Random Forest is an ensemble learning technique that combines multiple decision trees to create robust and accurate models. Predictions from individual trees are aggregated, which reduces overfitting and improves accuracy. They are robust against noisy data and outliers due to the averaging effect of multiple trees. Training individual trees can be parallelized for computational efficiency, making it faster than other methods. In the experiments, we configure Random Forest with six different decision trees, the Gini criterion to measure the quality of a split, and no limits for the depth of the trees.

Support Vector Machines (SVMs) excel in high-dimensional feature spaces and find optimal hyper-planes to separate different classes. In the experiments, we use two versions of SVM: a Linear SVM (SVM Linear) that finds a hyperplane that best separates classes in the feature space, following a one-vs-all approach and a Radial Basis Function SVM (SVM RBF) that handles nonlinear data by mapping it to a higher-dimensional space using kernel functions. The latter follows a one-vs-one approach so that each classifier differentiates between two classes. The decision boundary becomes a nonlinear surface in this case. In our experiments, we also use a small regularization parameter.

The time complexity of these algorithms varies depending on the dataset size and the number of features. Logistic Regression and Naive Bayes are less computationally expensive than SVMs and Random Forests, especially on large datasets. Logistic Regression and Naive Bayes are more memory-efficient and computationally less demanding, making them suitable for large datasets with many features. Random Forests and SVMs, particularly with non-linear kernels, are both memory and computationally intensive, especially as the size of the dataset grows. Random Forests require memory to store multiple trees, while SVMs require substantial memory for storing support vectors and the kernel matrix.

6. Metrics and Experimental Design

We use standard metrics to evaluate the performance of the methods. In particular, we rely on the *precision*, *recall*, and $F_1 - score$ for each label and classifier. Since our dataset is unbalanced, with certain labels over-represented—see Table A1—, we calculate the average for each label and then compute the average of the ratios for all the labels.

The metrics are based on the number of true positive classifications, TP, true negatives, TN, false positives, FP, and false negatives, FN. The *precision* represents the rate of positive predictions that are correctly classified and is given by the following expression:

$$precision = \frac{TP}{TP + FP}. \tag{3}$$

The *recall* is the rate of positive cases that the classifier correctly predicts and is calculated as:

$$recall = \frac{TP}{TP + FN}, \tag{4}$$

and the $F_1 - score$ is the harmonic mean of the *precision* and *recall*, given by the following:

$$F_1 - score = 2\frac{precision \cdot recall}{precision + recall}. \tag{5}$$

In the first part of the experiments, we establish a global ranking of the methods using these metrics with the test set. Then, we evaluate the performance for each template directory separately, which allows us to understand the behavior of each technique to data similar to the training set and data that are completely new, such as in the last template.

Choosing the best method, we rank the labels according to their precision range. We show the labels that are classified with high precision and those that are more difficult to classify. This will allow us to identify the labels that need to be considered for improving the method. We rely on confusion matrices to assess the errors produced between different labels. This complements the previous information and permits us to identify the labels usually classified as others. This can guide us in defining new features to differentiate between those labels.

In the last experiment, we study the influence of features on the accuracy of our method. We explore the benefit of the custom features with and without TF-IDF. We gradually increase the size of the context window by adding one word on the left and right of the central word. Therefore, we consider an odd number of words, ranging from one to eleven, i.e., from five to fifty-five features.

Implementation Details

The source code for the experiments was implemented in Python using the `scikit-learn` [38] and `Pandas` libraries. It is available in Github at https://github.com/jsanchezperez/electricity_invoice_extraction.git (accessed on 12 February 2024).

PDF files were converted into raw text using the `pdftotext` program. We removed strips of dots as they typically appear for horizontal lines during the conversion from PDF to text files. For convenience, we created a text file for each training and test directory, as explained in Section 3.2, each containing fifty different invoices.

These files were loaded using the `Pandas` library, separating the sentences and labels into two *dataframes*. The training data were split into two sets: 80% for training and 20% for validation. The validation set was used to evaluate the training process and tune the hyperparameters of the methods. Similarly, the test directory was loaded into another pair of *dataframes*. We carried out a final training step, where the whole training data—including the validation set—were used to train the models. This usually provides slightly better results. The test set was only used at the end to evaluate the models and report the results.

The hyperparameters of the models were tuned by hand. We tested default configurations and checked for the most relevant hyperparameters. In the case of SVM, we tested different kernels and various regularization parameters. Decision Trees and Random Forests were calculated without depth limit and with one sample at each leaf node. For the rest of the models, we tested various regularization parameters.

We created a pipeline that transformed the input data into a new *dataframe*, with the TF-IDF and custom features, and then called the corresponding machine-learning technique for fitting the classifier parameters to the training data.

In inference mode, the steps of the pipeline are as follows:

1. Convert the invoice document from PDF to text format;
2. Create a file with eleven-word sentences from the text file as explained in Section 3.2;
3. Compute TF-IDF features for each sentence;
4. Calculate custom features for the words of each sentence;
5. Concatenate both types of features;
6. Feed one of the machine-learning techniques with a *dataframe* containing these features.

The output is a new *dataframe* with the detected labels. In a post-processing step, a JSON file can be created using the detected labels and the corresponding central words of every sentence. Labels with multiple words can be easily extracted by concatenating consecutive values in the output data.

7. Results

In this section, we assess the performance of methods by comparing their *precision*, *recall*, and $F_1 - score$ on the test set. Table 5 compares global metrics for each method by first calculating the scores for each label and then computing the average, giving the same weight to each label.

Table 5. Global *precision*, *recall*, and $F_1 - score$ of each method, sorted by *precision*. The SVM method with Radial Basis Functions (SVM RBF) provides the best results, followed by Random Forest.

Method	Precision	Recall	F1-Score
Naive Bayes	61.41%	80.84%	66.11%
SVM Linear	80.74%	90.89%	84.55%
Logistic Regression	87.54%	81.74%	83.03%
Decision Tree	85.30%	83.08%	83.29%
Random Forest	91.58%	81.96%	85.47%
SVM RBF	91.86%	89.10%	89.63%

The result of Naive Bayes is poor compared with the other methods, with a recall significantly higher than its precision. Thus, this method detects many false positives but it allows detecting a larger rate of true positives. This method is fast, but its scores are low.

The performance of SVM Linear and Logistic Regression is much better, with a precision and recall higher than 80%. In the case of Logistic Regression, the recall is lower than the precision, so this method underestimates some labels. The Decision Tree yields a similar performance, with a balanced precision and recall.

Random Forest and SVM RBF provide similar results, with SVM RBF yielding the best performance of all methods. The main difference resides in a higher recall for SVM; thus, this method tends to detect a larger rate of positive cases.

The performance of each method for every template is displayed in Table 6. These results demonstrate the capacity of each one to adapt to new data. The precision is high for the first six templates but smaller for the last template, because the training data do not include samples from that template.

Logistic Regression is typically better than Decision Trees and SVM Linear for templates used during training. However, it is worse for invoices with new layouts and vocabulary. On the other hand, although SVM RBF provides the best results for documents already seen by the classifier, Random Forest yields better results for new types of documents. Therefore, the generalization capability of Decision Trees and Random Forests is better than the rest of the techniques. Nevertheless, we must note that the number of different templates in this dataset is not high and there is low variability in the training set.

Table 6. Precision for each template. The columns show the precision of the methods for each of the seven templates. The results for the first six templates are higher than the last template because documents of the latter were not used for training.

Method	T1	T2	T3	T4	T5	T6	T7
Naive Bayes	66.68%	65.46%	66.85%	60.92%	64.14%	64.45%	54.18%
SVM Linear	84.33%	84.91%	86.00%	83.58%	90.08%	88.71%	62.05%
Logistic Regression	90.50%	91.02%	93.54%	89.48%	95.41%	92.24%	61.03%
Decision Tree	91.35%	88.25%	88.63%	90.16%	92.12%	91.20%	65.86%
Random Forest	93.69%	**95.06%**	93.01%	92.61%	94.59%	93.80%	**68.56%**
SVM RBF	**94.49%**	94.26%	**97.82%**	**93.15%**	**98.19%**	**98.47%**	67.20%

Bold letters highlight the best result in each column.

7.1. Precision for Each Label

A closer examination of the results of the best model, SVM RBF, reveals that it achieves high precision for most labels, as shown in Table 7. The method provides a precision higher than 99% for 41% of the labels and higher than 90% for another 31%. This means that most labels are classified with high precision. Only 6% of labels have a precision smaller than 70%.

There is only one label with a precision smaller than 50%, `K2d`, which is related to the access toll rate. Looking at Table A1, we observe that this label appears 5000 times in the training set, i.e., in only one template. Label `K4d` is also classified with low precision and, similarly, appears in one template.

Table 7. SVM RBF: Classification of labels regarding their precision range. The first column stands for the precision range; the second column lists the labels in that range; the third column shows the percentage of labels in each group.

Precision	Labels	Percentage
[99–100%)	`B1`, `C7`, `CE`, `D9`, `E1`, `E3l`, `E5`, `E6`, `E7`, `E7p`, `E7s`, `E8`, `F2`, `F3`, `F3p`, `F3s`, `F4`, `F4p`, `F4s`, `F5p`, `F5s`, `F6`, `F7`, `G1`, `G2`, `G3`, `G3p`, `G6`, `G7`, `G8`, `G9`, `J4`, `K4`, `K6`, `KB`, `M3`, `M3d`, `N3`, `N4`, `N6`, `OO`	41.4%
[90–99%)	`A1`, `A3`, `A6`, `B2`, `B3`, `B4`, `B5`, `B6`, `C2`, `C3`, `C4`, `C5`, `C6`, `C8`, `C9`, `CB`, `E2`, `E4`, `E9`, `F1`, `G4`, `G5`, `GA`, `I3`, `J1`, `J2`, `J5`, `K3`, `N1`, `N2`, `N5`	31.3%
[80–90%)	`A4`, `A5`, `C1`, `CA`, `CC`, `CD`, `D1`, `DD`, `E3`, `F5`, `J3`, `K9`, `M4`, `N7`, `N8`	15.2%
[70–80%)	`F4u`, `F5u`, `F8l`, `I1`, `KA`, `KD`	6.1%
[60–70%)	`I2`, `K2`, `K5`, `KC`	4.0%
[50–60%)	`K4d`	1.0%
[0–50%)	`K2d`	1.0%

Analyzing the precision of other labels, we realize that it is related to the number of occurrences in the training set. Therefore, we may conclude that the method classifies the labels correctly if they appear in many documents of various templates.

Figure 2 depicts the confusion matrices for the SVM RBF method. Since the number of labels is too large, we have divided the matrix into six parts with eighteen labels each. We observe that the predictions consistently match the ground truths for most labels. The null values in the diagonal correspond to labels that do not appear in the test set.

One of the most notable confusions is between labels `k2d` and `k4d`, which are related to marketer prices. These amounts are similar, so this confusion may seem reasonable. Additionally, these labels are present in only one template of the training set. Another significant confusion is given by `F4u` and `F5u`, which represent the start and end of the billing period. These labels are similar and only appear in two templates. It is also interesting to note that the `OO` label, which occurs with high frequency, is hardly confused

with another label. In this sense, the method seems to learn patterns associated with out-of-class words correctly, which validates our strategy of integrating these types of words in the training data. This is not the case for other techniques where this label presents the largest mispredictions, such as Naive Bayes or SVM Linear. In these cases, it is necessary to implement several strategies to reduce the impact of imbalanced classes.

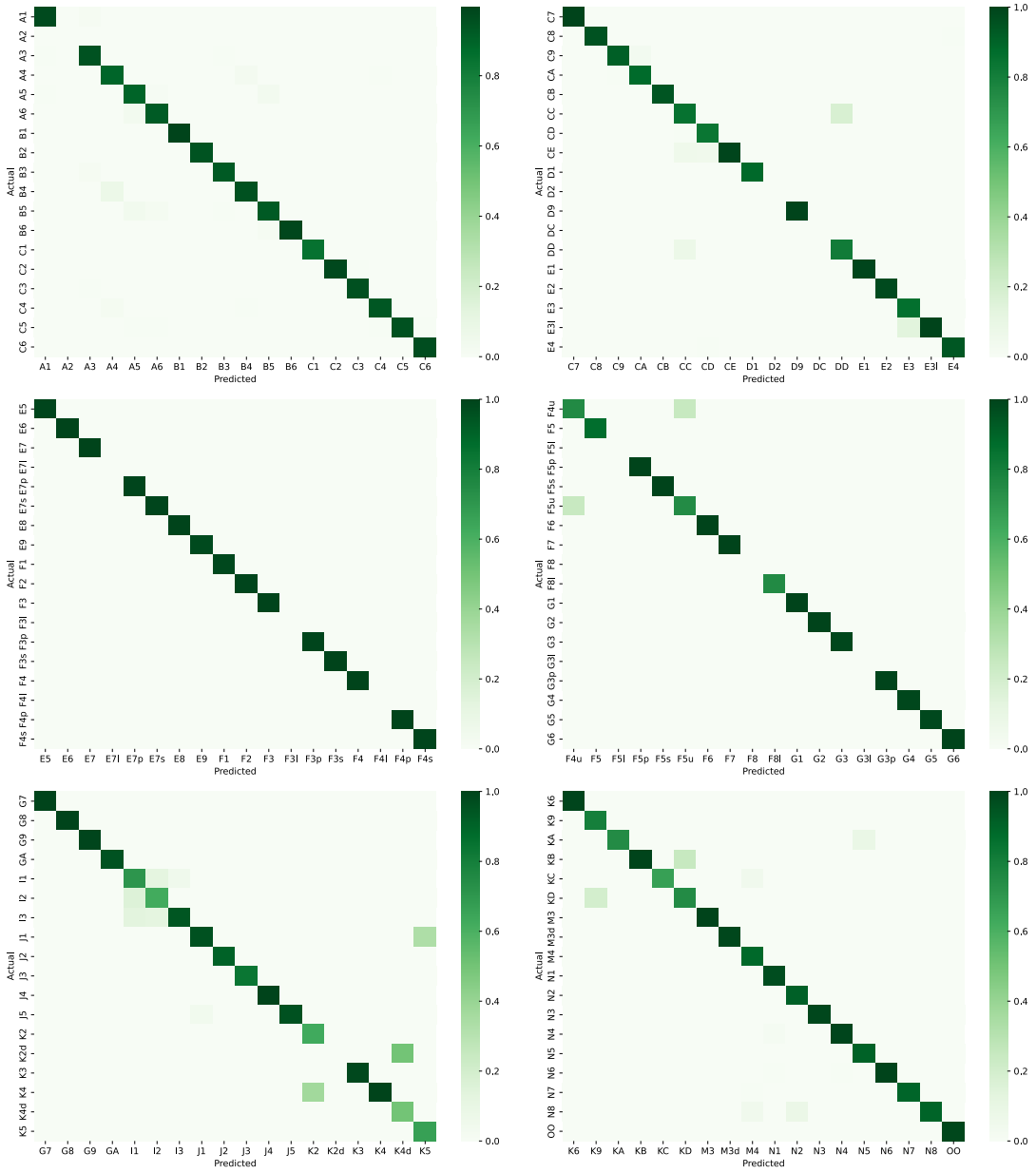

Figure 2. Confusion matrices for the SVM RBF classifier.

7.2. Analysis of Feature Influence

This section examines how the inclusion of different features impacts the precision of the methods. Our previous results were calculated using eleven words per sentence, yielding fifty-five custom features. We may reduce the number of words from zero to eleven.

In our first experiment, we compare the precision of the methods without using TF-IDF. Table 8 shows the results when we increase the number of custom features. If we only train with the central word, i.e., five custom features, the precision of all methods is too low, with Decision Tree and Random Forest providing slightly better results. The precision significantly augments with fifteen and twenty-five features, i.e., three and five words,

respectively, although it is not sufficient to obtain good performance. With seven to eleven words, Random Forest, SVM RBF, and Decision Tree yield good results, with the former consistently providing the best results.

Table 8. Analysis of custom features. Comparison of the precision of methods increasing the number of custom features. In this case, the training data does not include TF-IDF. In the second column, we show the results using five custom features (5-CF) for the central word; in the third column, we show the results for fifteen custom features (15-CF), corresponding to a context of three words and so on. The precision is very low for less than thirty-five features—seven words—, and for more features, Random Forest, SVM RBF, and Decision Tree provide good performance.

Methods	**5-CF**	**15-CF**	**25-CF**	**35-CF**	**45-CF**	**55-CF**
Naive Bayes	1.78%	16.69%	23.23%	27.56%	32.79%	34.25%
SVM Linear	2.96%	18.62%	29.63%	39.06%	48.14%	55.44%
Logistic Regression	3.65%	22.89%	43.20%	57.46%	65.30%	69.51%
Decision Tree	**4.33%**	**46.26%**	66.85%	77.50%	82.18%	84.46%
Random Forest	**4.33%**	46.22%	**67.02%**	**80.63%**	**85.44%**	**88.73%**
SVM RBF	3.65%	32.83%	61.61%	79.83%	84.02%	87.38%

Bold letters highlight the best result in each column.

Table 9 shows the precision of methods using TF-IDF with and without custom features. We observe that Logistic Regression provides the best results for TF-IDF, although the precision is low. The performance of other methods is poor, especially for SVM RBF. Random Forest provides the second-best result. Using five custom features significantly improves the precision in general.

Table 9. Analysis of the combination of TF-IDF and custom features. Comparison of the methods using TF-IDF and an increasing number of custom features. In the second column, we show the results of TF-IDF only; in the third column, we show the results of TF-IDF and five custom features (5-CF) for the central word; in the fourth column, we show the results of TF-IDF and fifteen custom features (15-CF), corresponding to a context of three words and so on. We observe that TF-IDF and our custom features are necessary to obtain high precision. SVM RBF and Random Forest provide the best results with TF-IDF and fifty-five custom features, respectively.

Methods	**TFIDF**	**5-CF**	**15-CF**	**25-CF**	**35-CF**	**45-CF**	**55-CF**
Naive Bayes	36.19%	54.47%	64.45%	65.08%	62.54%	61.34%	61.41%
SVM Linear	35.10%	59.57%	71.44%	77.00%	79.40%	79.69%	80.74%
Logistic Regression	**56.79%**	**76.32%**	82.49%	85.94%	87.03%	87.30%	87.54%
Decision Tree	46.95%	71.83%	81.15%	81.65%	84.19%	84.83%	85.30%
Random Forest	51.73%	73.10%	83.14%	87.11%	89.13%	91.42%	91.58%
SVM RBF	26.30%	73.79%	**86.86%**	**89.55%**	**90.74%**	**91.48%**	**91.86%**

Bold letters highlight the best result in each column.

Logistic Regression remains the top-performing method in this scenario. However, the gap with other methods, particularly Random Forest and SVM RBF, has narrowed considerably. These latter methods show a significant improvement in precision. Using more custom features allows for improving the results of the methods, except for Naive Bayes. SVM RBF provides the best results in these cases, followed by Random Forest, Logistic Regression, and Decision Tree.

Comparing the results in Tables 8 and 9, we may conclude that TF-IDF is important when the number of custom features is small and is necessary to attain high precision. SVM RBF, Random Forest, and Decision Tree provide good results if TDF-IF is not used. Decision Tree yields similar results in both cases. It is worth mentioning that Random

Forest is slightly better than SVM RBF when not using TF-IDF and SVM RBF is better when we use it.

TF-IDF is much more important for Naive Bayes and Logistic Regression, which obtains the greatest gain in precision from these features, ranking in the third position before the Decision Tree.

This information is also represented in the graphics of Figures 3 and 4, respectively. SVM RBF and Random Forest are the best methods for a given set of features in both configurations, followed by Decision Tree. Logistic Regression and Naive Bayes perform much better with TF-IDF. In this setting, Logistic Regression outperforms Decision Tree.

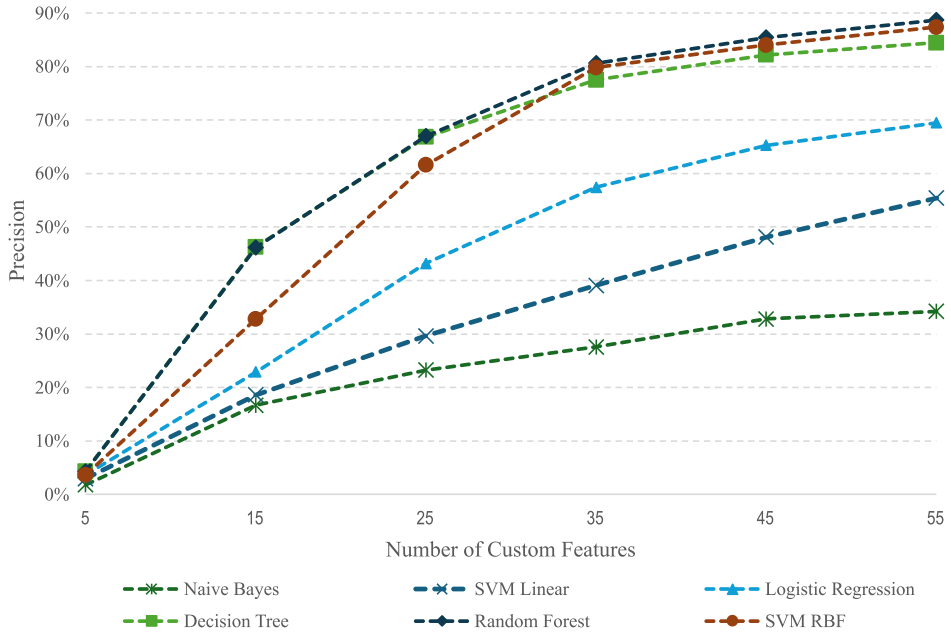

Figure 3. Influence of custom features. This graphic shows the influence of increasing the number of custom features without relying on TF-IDF.

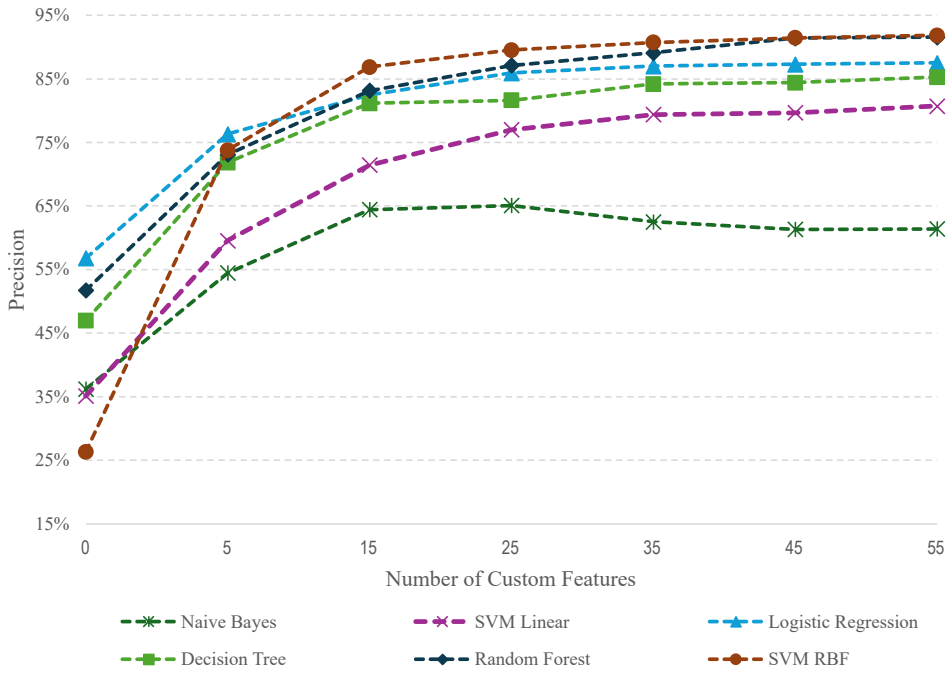

Figure 4. Influence of custom features and TF-IDF. This graphic shows the influence of increasing the number of custom features with TF-IDF.

8. Discussion

In this work, we proposed a methodology to extract relevant information from electricity invoices based on text data. We converted the annotated bills, originally in PDF format, into text files. These files were then pre-processed to generate tuples of sentences and their associated labels. Each sentence contained eleven words, with the label associated with the central word. Our method employed a bag-of-words strategy and a set of custom-designed features, specially designed for electricity data. We assessed the performance of various machine learning algorithms.

Our results demonstrated that TF-IDF features, used in the bag-of-words strategy, were key for obtaining an acceptable precision. They also showed that our custom features were necessary to improve the results. We demonstrated that the number of words used to calculate features was also important, with more than seven being sufficient to attain high precision in general.

We studied the performance of various standard machine-learning techniques and found that Naive Bayes did not provide good results, but its precision significantly increased with TF-IDF, although still insufficient. Logistic Regression was the method that benefited the most from TF-IDF.

The techniques that provided the best results were SVM RBF and Random Forest. The latter yielded the best precision with custom features alone. SVM RBF, on the other hand, provided the best results when both TF-IDF and custom features were employed. Logistic Regression stood out from the others using TF-IDF and five custom features. However, TF-IDF alone was not enough to tackle this problem for any method. We may conclude that TF-IDF was important as a baseline for all the methods but the custom features were necessary to achieve higher precision.

Looking at the classification rate of SVM RBF, we observed that most labels were classified with a precision higher than 90% and only two labels obtained a precision lower than 60%. The precision obtained for the `OO` label validates our strategy of integrating out-of-class words into the training process. We believe that this approach is flexible enough to generalize to new words not belonging to any predefined class. Since our model discriminates based on features extracted from the word and its context, the likelihood of out-of-class words coinciding with one of the predefined classes must be low.

We also assessed the performance of the methods for each template. The precision for templates used during training was significantly higher. In this case, SVM RBF provided the best results and Random Forest was better for unseen invoices. The difference in precision for seen and unseen data was bigger than 30%, which indicates some overfitting to the training data.

This is reasonable because the number of templates in this dataset is small. The variety of data is not high since each template shares a common vocabulary and layout.

Therefore, one strategy to improve the performance of these methods is to increase the number and variety of templates in the dataset. Increasing the number of custom features at the word level is unlikely to improve the results since the precision of most methods does not significantly improve after seven words. It would be more interesting to explore new features related to the position of words inside the documents and their relative position to other words and labels.

This work has several limitations. For example, due to the small diversity of documents in the dataset, the generalization capacity of our method is limited and may produce unsatisfactory results for new invoices. We also note that the dataset contains region-specific invoices and, therefore, the performance of our method will be poor for invoices from other regions.

Regarding the training of the methods, we did not conduct an extensive search for the best hyperparameters. We tuned the hyperparameters by hand, leveraging our domain expertise and understanding of the models. Consequently, the results of each technique may provide slightly better results for other configurations, at the expense of a considerable

increase in training and validation time. However, we do not expect a significant change in the results of our study.

In this study, we assessed the performance of our method using several machine-learning approaches, including Logistic Regression and Naive Bayes which can be seen as baseline models. These models are chosen due to their widespread use and effectiveness in similar tasks, providing a robust benchmark for evaluating our approach. These well-established models provided a comprehensive comparison, highlighting the advantages of our approach. It would have been interesting to compare with other state-of-the-art models, but adapting such methods to our dataset requires a significant effort. Future research can build on our results to compare with new or existing methods, further validating our approach.

Our method can enhance the automation of information extraction from electricity invoices, leading to more efficient business processes and reduced manual labor. This can benefit utility companies, marketers, and distributors by improving data accuracy and accelerating customer service response times. This scalable solution can easily be adapted to diverse invoice formats and contents.

9. Conclusions

This work analyzed the performance of various machine-learning techniques for extracting many fields from electricity invoices. These are complex documents that include many different contents and layouts. We relied on a bag-of-words strategy and custom-designed features tailored for these invoices.

We demonstrated that combining TF-IDF and our custom features was necessary to obtain high precision in several methods. The techniques that provided the best results were SVM RBF and Random Forest, although the results of the Decision Tree were also competitive. Logistic Regression was better than Decision Tree when we employed TF-IDF features. We also showed that the precision for already-seen template documents was higher than for new templates. This is reasonable as the dataset contains a reduced number of templates.

In future works, we will increase the number of templates in the dataset. This is important for augmenting the variability of documents with richer vocabulary and layout. We will explore new features based on the position of words and the relative spatial distribution of labels. We are also interested in using deep learning techniques, especially those based on Large Language Models and multimodal architectures.

Author Contributions: Conceptualization, J.S.; methodology, J.S.; software, J.S. and G.A.C.-L.; validation, J.S. and G.A.C.-L.; formal analysis, J.S.; data curation, G.A.C.-L.; writing–original draft preparation, J.S.; writing—review and editing, J.S. and G.A.C.-L.; supervision, J.S.; funding acquisition, J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Servicio Canario de Salud, Gobierno de Canarias, under the grant number F2022/03.

Informed Consent Statement: Not applicable.

Data Availability Statement: The IDSEM dataset (Sánchez et al. [13]) is publicly available in Figshare at https://doi.org/10.6084/m9.figshare.c.6045245.v1 (accessed on 12 February 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Network
LSTM	Long Short-term Memory
NER	Named Entity Recognition
NLP	Natural Language Processing
OCR	Optical Character Recognition
SVM	Support Vector Machine

Appendix A. Number of Occurrences per Label

Table A1 details the number of times each label appears in the invoices of the training set of the IDSEM dataset.

There are six directories with 5000 invoices each, making 30,000 documents. The table shows that various labels appear more than 30,000 times, meaning that they appear several times in some documents or the label contains a chunk of words. For example, code `C1`, corresponding to the Marketer's name, appears 564,700 times in the bills. This means the field is included several times in the documents and contains multiple words. Other labels may appear several times in the same document, such as label `J2`, which stands for the total price, but with a single token.

On the contrary, some labels appear less than 30,000 times, meaning that some documents do not include that information. For instance, the invoice reference number, `F2`, only appears in four directories, and the marketer's province, `C6`, is present in one directory. A few labels do not appear in any template, but these are typically related to dates in a large format, such as `E7l`, `F3l`, or `F4l`.

Table A1. Number of times that each label appears in the invoices of the training set. There are 30,000 documents divided into six directories, each containing 5000 bills. Some labels appear multiple times in several template documents, such as `C7`, `J5`, or `M4`, whereas others only appear in a few templates, such as `B6`, `C6`, or `G1`.

Label Code	Number of Occurrences
`OO`	30,794,100
`C7`	599,600
`C8`	593,100
`C1`	564,700
`E2`	210,400
`D1`	181,400
`CB`	151,200
`G2`	150,000
`B3`	149,200
`C9`	143,600
`A3`	129,500
`G3`	125,000
`C3`	123,700
`GA`	121,100
`CC`	105,600
`J5`	105,000
`F3`	100,000
`A1`, `B1`	91,600
`M4`	90,000
`I3`	75,500
`B5`	71,700
`F6`	65,000
`A5`	64,500
`CD`	64,100
`N4`, `N6`	60,000
`CA`	55,600
`J3`	55,000
`C5`	54,400
`C2`	50,900
`E7`,`F4`, `F5`, `J2`,	50,000

Table A1. *Cont.*

Label Code	#Occurrences
CE	45,500
E3, N2, N5	45,000
N8, N7	44,700
J1	40,300
N1	40,000
A4, G4	35,000
C4	33,200
DD	31,100
B2, B4, D9, E1, E3l, E4, E6, E8, E9, F1, F4p, G9, I1, I2, J4	30,000
A6	28,000
F4s, F5s, M3d	25,000
F5p	20,500
F2, KD, K6	20,000
G1	19,900
B6	16,300
E5, K9, F8l, KA	15,000
E7s, E7p, F7, F5u, F3p, F4u, G5, KC, KB, K2, K3, K5, N3	10,000
G3p	9700
C6, F3s, G6, G7, G8, K2d, K4d, K4, M3	5000
A2, D2, DC, E7l, F3l, F4l, F5l, F8, G3l	0

References

1. Dengel, A.R.; Klein, B. smartFIX: A Requirements-Driven System for Document Analysis and Understanding. In *Proceedings of the Document Analysis Systems V*; Lopresti, D., Hu, J., Kashi, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2002; pp. 433–444.
2. Huang, Z.; Chen, K.; He, J.; Bai, X.; Karatzas, D.; Lu, S.; Jawahar, C.V. ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction. In Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, 20–25 September 2019; pp. 1516–1520. [CrossRef]
3. Dhouib, M.; Bettaieb, G.; Shabou, A. DocParser: End-to-end OCR-Free Information Extraction from Visually Rich Documents. In Proceedings of the Document Analysis and Recognition—ICDAR 2023, San José, CA, USA, 21–26 August 2023; Fink, G.A., Jain, R., Kise, K., Zanibbi, R., Eds.; Springer: Cham, Switzerland, 2023; pp. 155–172.
4. Rusiñol, M.; Benkhelfallah, T.; d'Andecy, V.P. Field Extraction from Administrative Documents by Incremental Structural Templates. In Proceedings of the 2013 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013; pp. 1100–1104. [CrossRef]
5. Shapenko, A.; Korovkin, V.; Leleux, B. ABBYY: The digitization of language and text. *Emerald Emerg. Mark. Case Stud.* **2018**, *8*, 1–26. [CrossRef]
6. Schuster, D.; Muthmann, K.; Esser, D.; Schill, A.; Berger, M.; Weidling, C.; Aliyev, K.; Hofmeier, A. Intellix–End-User Trained Information Extraction for Document Archiving. In Proceedings of the 2013 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 101–105.
7. Holt, X.; Chisholm, A. Extracting structured data from invoices. In Proceedings of the Australasian Language Technology Association Workshop 2018, Dunedin, New Zealand, 10–12 December 2018; pp. 53–59.
8. Yadav, V.; Bethard, S. A Survey on Recent Advances in Named Entity Recognition from Deep Learning Models. *arXiv* **2019**, arXiv:1910.11470.
9. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. Neural Architectures for Named Entity Recognition. *arXiv* **2016**, arXiv:1603.01360.
10. Salgado, A.; Sánchez, J. Information extraction from electricity invoices through named entity recognition with Transformers. In Proceedings of the 5th International Conference on Advances in Signal Processing and Artificial Intelligence, Tenerife, Spain, 7–9 June 2023; International Frequency Sensor Association (IFSA) Publishing: Indianapolis, IN, USA, 2023; pp. 140–145. [CrossRef]
11. Cheng, Z.; Zhang, P.; Li, C.; Liang, Q.; Xu, Y.; Li, P.; Pu, S.; Niu, Y.; Wu, F. TRIE++: Towards End-to-End Information Extraction from Visually Rich Documents. *arXiv* **2022**, arXiv:2207.06744.
12. Huang, Y.; Lv, T.; Cui, L.; Lu, Y.; Wei, F. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking. In Proceedings of the MM'22: 30th ACM International Conference on Multimedia, New York, NY, USA, 10–14 October 2022; pp. 4083–4091. [CrossRef]

13. Sánchez, J.; Salgado, A.; García, A.; Monzón, N. IDSEM, an Invoices Database of the Spanish Electricity Market. *Sci. Data* **2022**, *9*, 786. [CrossRef] [PubMed]
14. Manning, C.D.; Raghavan, P.; Schütze, H. *Introduction to Information Retrieval*; Cambridge University Press: Cambridge, UK, 2008.
15. Park, S.; Shin, S.; Lee, B.; Lee, J.; Surh, J.; Seo, M.; Lee, H. CORD: A consolidated receipt dataset for post-OCR parsing. In Proceedings of the Workshop on Document Intelligence at NeurIPS 2019, Vancouver, BC, Canada, 14 December 2019.
16. Šimsa, Š.; Šulc, M.; Uřičář, M.; Patel, Y.; Hamdi, A.; Kocián, M.; Skalickỳ, M.; Matas, J.; Doucet, A.; Coustaty, M.; et al. DocILE Benchmark for Document Information Localization and Extraction. In Proceedings of the International Conference on Document Analysis and Recognition, San José, CA, USA, 21–26 August 2023; Springer: Cham, Switzerland, 2023.
17. Yu, W.; Zhang, C.; Cao, H.; Hua, W.; Li, B.; Chen, H.; Liu, M.; Chen, M.; Kuang, J.; Cheng, M.; et al. ICDAR 2023 Competition on Structured Text Extraction from Visually-Rich Document Images. In Proceedings of the Document Analysis and Recognition—ICDAR 2023, San José, CA, USA, 21–26 August 2023; Fink, G.A., Jain, R., Kise, K., Zanibbi, R., Eds.; Springer: Cham, Switzerland, 2023; pp. 536–552.
18. Esser, D.; Schuster, D.; Muthmann, K.; Berger, M.; Schill, A. Automatic indexing of scanned documents: A layout-based approach. In Proceedings of the Document Recognition and Retrieval XIX, Burlingame, CA, USA, 22–26 January 2012; Viard-Gaudin, C., Zanibbi, R., Eds.; International Society for Optics and Photonics (SPIE): Bellingham, WA, USA, 2012; Volume 8297, pp. 118–125. [CrossRef]
19. d'Andecy, V.P.; Hartmann, E.; Rusiñol, M. Field Extraction by Hybrid Incremental and A-Priori Structural Templates. In Proceedings of the 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), Vienna, Austria, 24–27 April 2018; pp. 251–256. [CrossRef]
20. Goldberg, Y. *Neural Network Methods for Natural Language Processing*; Springer Nature: Cham, Switzerland, 2022.
21. Ma, X.; Hovy, E. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. *arXiv* **2016**, arXiv:1603.01354.
22. Hamdi, A.; Carel, E.; Joseph, A.; Coustaty, M.; Doucet, A. Information Extraction from Invoices. In Proceedings of the Document Analysis and Recognition—ICDAR 2021, Lausanne, Switzerland, 5–10 September 2021; Lladós, J., Lopresti, D., Uchida, S., Eds.; Springer: Cham, Switzerland, 2021.
23. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
24. Katti, A.R.; Reisswig, C.; Guder, C.; Brarda, S.; Bickel, S.; Höhne, J.; Faddoul, J.B. Chargrid: Towards Understanding 2D Documents. *arXiv* **2018**, arXiv:1809.08799.
25. Zhao, X.; Niu, E.; Wu, Z.; Wang, X. CUTIE: Learning to Understand Documents with Convolutional Universal Text Information Extractor. *arXiv* **2019**, arXiv:1903.12363.
26. Palm, R.B.; Winther, O.; Laws, F. Cloudscan—A configuration-free invoice analysis system using recurrent neural networks. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9–15 November 2017; IEEE: Piscataway, NJ, USA, 2017; Volume 1, pp. 406–413.
27. Kuang, J.; Hua, W.; Liang, D.; Yang, M.; Jiang, D.; Ren, B.; Bai, X. Visual Information Extraction in the Wild: Practical Dataset and End-to-End Solution. In Proceedings of the Document Analysis and Recognition—ICDAR 2023, San José, CA, USA, 21–26 August 2023; Fink, G.A., Jain, R., Kise, K., Zanibbi, R., Eds.; Springer: Cham, Switzerland, 2023; pp. 36–53.
28. Zhang, P.; Xu, Y.; Cheng, Z.; Pu, S.; Lu, J.; Qiao, L.; Niu, Y.; Wu, F. TRIE: End-to-End Text Reading and Information Extraction for Document Understanding. In Proceedings of the MM'20: 28th ACM International Conference on Multimedia, New York, NY, USA, 12–16 October 2020; pp. 1413–1422. [CrossRef]
29. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2019; Volume 32, pp. 5753–5763.
30. Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.; Salakhutdinov, R. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; Korhonen, A., Traum, D., Màrquez, L., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 2978–2988. [CrossRef]
31. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Los Alamitos, CA, USA, 11–17 October 2021; pp. 9992–10002. [CrossRef]
32. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
33. Denk, T.I.; Reisswig, C. BERTgrid: Contextualized Embedding for 2D Document Representation and Understanding. *arXiv* **2019**, arXiv:1909.04948.
34. Xu, Y.; Li, M.; Cui, L.; Huang, S.; Wei, F.; Zhou, M. LayoutLM: Pre-Training of Text and Layout for Document Image Understanding. In Proceedings of the KDD'20: 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 6–10 July 2020; pp. 1192–1200. [CrossRef]

35. Tu, Y.; Guo, Y.; Chen, H.; Tang, J. LayoutMask: Enhance Text-Layout Interaction in Multi-modal Pre-training for Document Understanding. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Toronto, ON, Canada, 9–14 July 2023; Rogers, A., Boyd-Graber, J., Okazaki, N., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2023; pp. 15200–15212. [CrossRef]
36. Luo, C.; Cheng, C.; Zheng, Q.; Yao, C. GeoLayoutLM: Geometric Pre-training for Visual Information Extraction. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023.
37. Sánchez, J.; Salgado, A.; García, A.; Monzón, N. IDSEM Dataset. *Figshare* **2022**. [CrossRef]
38. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.