Tool for Automated Tax Coding of Invoices

Tarun Tater[1], Sampath Dechu[1], Neelamadhav Gantayat[1], Meena Guptha[2], Sivakumar Narayanan[2]
[1]IBM Research
[2]IBM Services
[ttater24, sampath.dechu, neelamadhav, meenamga, sivakumar.narayanan]@in.ibm.com

Abstract

Accounts payable refer to the practice where organizations procure goods and services on credit which need to be reimbursed to the vendors in due time. Once the vendor raises an invoice, it undergoes through a complex process before the final payment. In this process, tax code determination is one of the most challenging steps, which determines the tax to be levied and directly influences the amount payable to a vendor. This step is also very important from a regulatory compliance standpoint. However, it is error-prone, labor (resource) intensive, and needs regular training of the resources as it is done manually. Further, an error in the tax code determination induces penalties on the organization. Automatically arriving at a tax-code for a given product accurately and efficiently is a daunting task. To address this problem, we present an automated end-to-end system for tax code determination which can either be used as a standalone application or can be integrated into an existing invoice processing workflow. The proposed system determines the most relevant tax code for an invoice using attributes such as item description, vendor details, shipping and delivery location. The system has been deployed in production for a multinational consumer goods company for more than 6 months. It has already processed more than $22k$ items with an accuracy of more than 94% and high confidence prediction accuracy of around 99.54%. Using this system, approximately 73% of all the invoices require no human intervention.

Introduction

Accounts payable is a critical business process as it involves the majority of a company's bills and invoice payments. Also, it is important to efficiently manage the process such that the invoices are paid on time, accurately and avoiding penalties to ensure long-term relationship with the vendors (Schaeffer 2002).

One of the major steps while processing the invoices is determining the right tax code associated with the applicable tax percentage to make sure that the accurate liability is reported to the tax authorities. These are indirect taxes that is, consumption taxes which are levied on commodities or services before they reach the consumer. These are essentially paid by the consumer as part of the market price before being paid to the government. They are known as Value Added Tax (VAT) (Tait 1988; Keen and Lockwood 2010; Schenk, Thuronyi, and Cui 2015) or Goods and Services Tax (GST) (Khurana and Sharma 2016) or sales tax as per the specific country's regulations.

Interestingly, indirect taxes are imposed on sales of goods by businesses at each stage of production and distribution (Decoster et al. 2010; Anderson, De Palma, and Kreider 2001; Little 1951). Tax code determination becomes complex as there can be variable number of steps and checks for different goods and services. The objective of levying accurate tax is achieved by an input/output system enabling the following : (i) when a business operating in a VAT/GST country buys goods or services, it pays tax to the supplier which is called an *input* tax (Newbery 1986); (ii) when the same business sells goods or services to another business or to a final consumer, it is required to charge tax which is called as *output* tax; (Hoff 1991) (iii) the business must periodically sum up the input tax and deduct it from the output tax and pay the difference of the output tax to the government agency responsible for collecting it (Agha and Haughton 1996; Keen and Lockwood 2010).

Motivation

The challenges faced by the clients in tax determination (Hoppe et al. 2019, 2018; Devereux 2016) are further amplified by the fact that the tax regulations go through lot of changes over time. The breaking obstacle is that the compliance of taxes is governed by the regulations of the country and incorrect tax filed or noncompliance may lead to interest, monetary penalties, or prosecution as per specific country laws. Since compliance is mandatory, this complex manual labour intensive process of tax code identification consumes a lot of time and slows down the entire invoice processing flow (Hoppe et al. 2018; Ingraham and Karlinsky 2005; McKerchar 2005; Spengel, Evers, and Zinn 2012). There is every possibility for any of the novice processor to make a mistake in tax code determination. This would result in the invoice processor to pay the tax again under the correct code and then to apply for refund for the tax wrongly paid within the limitation period. The current process also requires regular training for existing and new resources.

One of the major tasks in a Accounts Payable system is to determine the correct tax code, as some fraudulent vendors try to raise false invoices with wrong tax codes (Keen

Item Description	Company code	Destination City	Destination Country	Origin Country	Vendor	Tax code
chg - leasing costs - 2019 - italy	IT15	napoli	it	us	v2	7N
new air treatment pilot plant	5262	cinisello	it	it	v1	7N
cabinets oristano feb 20	500S	oristano	it	de	v3	7O
1555350 cavalletto portacartelli	IT10	milano	it	it	v1	7N
prodtti spaccio iva 10%	IT15	roma	it	it	v2	ID
carta office depot everyday A4 80 gm? b	500S	assago	it	it	v1	7“

Table 1: Examples of invoice items including the input data and the output tax codes for tax code determination process in Italy.

and Smith 2006). While processing an invoice received by the vendor, the accounts payable team is expected to determine the applicable tax code on the basis of the expense or services received and certain additional parameters like origin country, destination city and country, and certain vendor details. This on first thought might seem like an easily automatable and scalable process once implemented for a specific client. However, as each country has its own rules and parameters for determining the tax, scalability is a key challenge. Currently the invoice processing team maintains all parameters and changes in a cheat sheet to refer while processing, however due to complexity of the process, there are errors and it requires various levels of quality check to make sure right tax is applied. In this research, we aim to automate the complex service process of correct tax code assignment by using machine leaning modules to reduce errors and make the process more efficient, which in turn saves the resources as well as avoid penalties thus having higher client engagement and satisfaction.

Contributions

In this paper, we present a configurable end to end system for tax code determination which improves over time because of the built-in feedback mechanism. The proposed system has the following key aspects:

1. A hybrid classifier which is a combination of semantic similarity module and a rule based engine with varying importance to each of them depending on the business needs. The proposed hybrid classifier outperforms different machine learning models trained on the historical data.
2. The semantic similarity system helps to associate the relevant item descriptions from historical data with the current item description. This is followed by the rule-engine which chooses the descriptions with the matching geographical and vendor details. Thereafter, the similar shortlisted data-points are sorted according to dates to tackle any possible drifts which inherently gives more importance to the recent data.
3. The system has a unique feedback mechanism which ensures that the system can tackle the concept and data drift. The feedback module also helps to change the confidence thresholds for which the feedback is required, hence improving the efficiency of the system and saving workhours. The feedback mechanism is designed to incorporate inter-agent agreement before fine-tuning the model with the received feedback.

We evaluate the proposed system by deploying it for a large global consumer goods company's invoices pertaining to Italy. It has been deployed and used successfully for over six months, which highlights the usefulness of the system.

Re-imagine the Accounts Payable Process

While processing an invoice received by the vendor, the accounts payable department is expected to determine the tax code from the Enterprise Resource Planning (ERP) system. The applicable tax is based on the nature of expense or services received, origin country, destination city and country, and any vendor exemptions.

In the current business-as-usual, when an invoice is received, the accounts payable agent validates if all required information is available on the invoice. After this, the agent performs three way matching between purchase order, invoice and goods received, or assign accounting codes to process the invoice in system, along with determining the tax code. The agent follows the following steps:

1. Determine the nature of expense based on the description on the invoice. In case, if the processor is unable to determine the category of expense based on the description of the invoice, agent maneuvers through several different documents and past data manually to find the broader category of expense.
2. Then, based on the shipping address, including the origin country, destination city and country, and nature of expense, the agent checks the documents available for the applicable tax codes. If this is unavailable, the agent then maneuvers through the previous transactions that had similar combinations to arrive at the applicable tax code. This is quite complex and time consuming when no exact matching descriptions are found since it involves a lot of data points with varying tax codes when examining the historical data.
3. The agent then checks if there is any additional text provided on the invoice that changes the value of tax code. After this, the agent checks if the vendor has tax exemptions.

Once all these validations are done, the appropriate tax code is updated for each item in the invoice, which is then processed for payment.

Automated Process - The automated tax code determination system has streamlined the above process. When an invoice is received, the accounts payable agent will validate

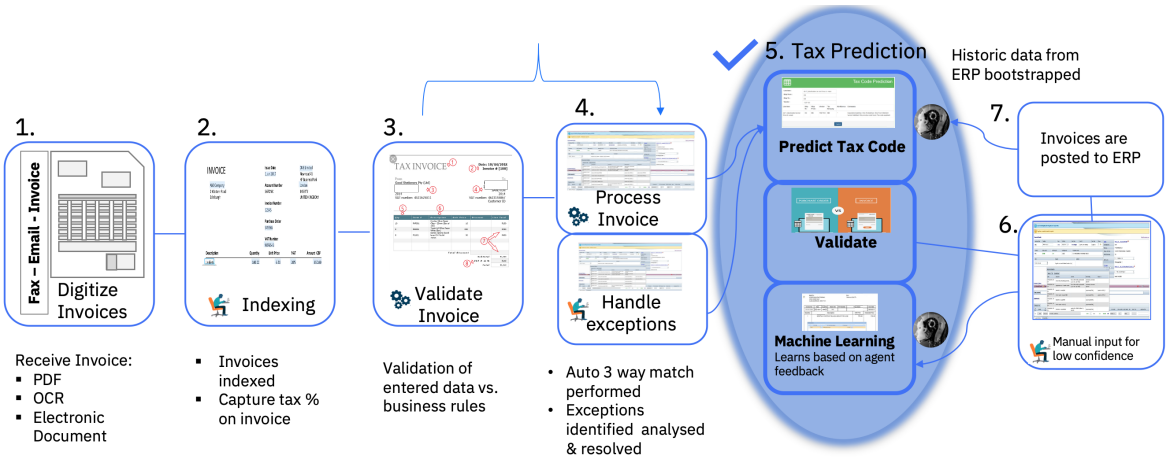

Figure 1: Invoice process re-imagined using proposed system including all the steps of invoice processing.

if all required information is available on the invoice, after which the agent would perform a three-way matching or assign accounting codes to process the invoice in system. The determination of tax code is completely automated with the manual intervention limited for providing feedback. The historical data pertaining to tax code determination by the human agents is uploaded as the training data. The tax module picks the description of the invoice along with the shipping addresses, including origin and destination countries, company code details, and vendor details and uses the model trained on the training data to determine the tax code. Each parameter is assigned a certain weight and the confidence scores are computed accordingly. When tax code is determined by the system, it also provides the confidence score to decides whether human intervention is required or not.

System Architecture

The overall system architecture is depicted in Figure 2. The system is trained using the historical data, which comprises of the item description, vendor name, origin country, destination city and country, company code, and posting date.

Once an invoice is indexed, each individual item is sent as an input to the tax code determination system. The first step is the semantic similarity engine searches for matching item description with descriptions in historical data above a threshold. As discussed in (Maurya et al. 2020), since item descriptions are not well-formed sentences, computing semantic similarity is a daunting task. The matching descriptions are then searched over the historical data to pool all the data points with these descriptions. These collected samples are filtered based on the matching shipping addresses and company codes (step 2 and 3 in Figure 2). The filtered results are then matched for vendor in step 4. If there are no vendor matches, this criteria is relaxed. The results are then sorted based on time (refer to step 5 in Figure 2) to adhere to the fact that tax code might have changed over time for provided details. The tax code is predicted by majority voting of the top-N results. The confidence of prediction (C_p) depends on various factors and is given by :

$$C_p = D_s * W_d + V_s * W_v + M_s$$

where D_s denotes scores for description similarity and is in the range of $0 < D_s < 1$ for exactly matching description. By design D_s for fuzzy matches would be a lower value as compared to exactly matching descriptions in a particular configuration. W_d is the configurable weight given for descriptions. V_s denotes the similarity score for vendor details, and W_v is the configurable weight given for vendor details. The majority voting score (M_s) is calculated as:

$$M_s = W_m * \frac{N_m}{N} * \frac{N_m}{N_T}$$

where W_m is the configurable weight given for popularity of tax code. N refers to the number of shortlisted data samples which are being considered for tax code classification. N_m describes the number of items with the majority tax code out of the shortlisted candidate samples. N_T describes the number of distinct tax codes in the N data samples. The matching score is so defined to adjust confidence giving importance to the majority tax code. Thus, if all the shortlisted candidates have the same tax code, it would get a higher confidence when compared to when there are two or more contesting tax codes. It is also important to note that -

$$W_m + W_s + W_v + W_t = 1$$

where W_t is the weight (importance) assigned to the tax percentage listed on the invoice.

According to the business needs, there are two thresholds of confidence which decides the flow of each item: (i) minimum confidence(C_{min}) and (ii) maximum confidence(C_{max}). If the confidence is above C_{max}, the item is assigned the predicted tax code and posted ahead without any manual intervention. If the predicted confidence(C_p) is between C_{max} and C_{min}, that is $C_{min} < C_p < C_{max}$, the item is then sent for feedback by agent. The agent sees the invoice copy along with the details of items and the predicted tax code. The agent can then choose

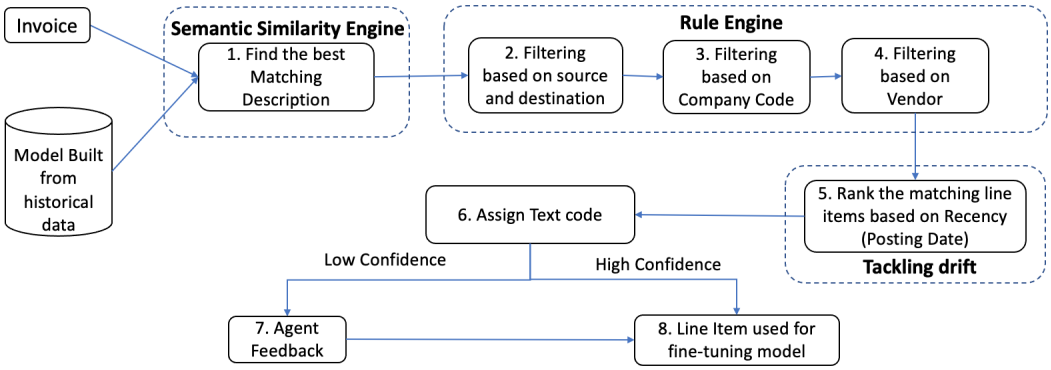

Figure 2: Architecture of proposed system involving Semantic Similarity Engine, Rule Engine, module for tackling Drift, Feedback Mechanism and Online training

to give an upvote, implying that the assigned tax code is correct, or give a downvote to choose the most suitable tax code. This feedback is used to enhance the model. In case the predicted confidence is lower than the minimum defined threshold, i.e. $C_p < C_{min}$, the item is sent back to the workflow for manual entry of tax code.

When the system is deployed for a new client, the confidence thresholds (C_{max} and C_{min}) would be conservative such that there will be a high C_{max} and lower value of C_{min}. This would be a hyper care period where for a few weeks the agents can validate the predictions. Analyzing the agent feedback over time, these thresholds are adjusted so as to maintain a very high accuracy rate ($> 99\%$) and at the same time reducing the manual effort of feedback. The lowering of C_{min} would help in determining more items which can be pushed for feedback instead of having to do a complete manual process by lowering the C_{min} threshold. It would also benefit the system since now it would have more feedback data. On the other hand, the lowering of C_{max} would result in more and more items being auto-posted. However, this needs to be done while maintaining high accuracy and reduce the manual work of feedback by agents.

The feedback captured about the tax code also includes the agent details. Once the system has enough data-points, the feedback collected can be used for better training of agents on a team and individual level.

When the system is not confident of the predictions, it throws an *Unknown* as the predicted tax code with zero confidence. This may happen because of no good matches for descriptions, or if the filtering produces zero shortlisted candidates. Another scenario for this is when the shortlisted candidates have conflicting tax codes with equal M_s. The idea behind having *Unknown* predictions is to reduce erroneous predictions. This helps improve system accuracy by reducing the error rates and boosts confidence among agents about the system as well. This approach is taken because it is better to not predict rather than wrong predictions.

Component Details

In this section, we will discuss the different components in detail including the data used for building and finalizing the tax code determination engine design.

Training Data Characteristics

We received the historical data of tax codes corresponding to different items in invoices ERP tool. It had details about the item such as the descriptions, shipping addresses (origin and destination), vendor details, posting dates, tax code assigned. The training data was collected for a time period of 14 months starting March 2018 to May 2019. It comprised of around $\sim$53,000 items each having 35 fields from invoices of Italy. The data pre-processing pipeline consisted of standardizing the data points, followed by tackling missing data by filling blank values as unknown or N/A. Since the tax codes have a very skewed distribution, we also employed oversampling techniques and classifiers with class weights. One of the other interesting challenges to tackle in the data was duplicate postings for the same invoice. This happens since human agents may have determined a wrong tax code, which would have been later corrected in the audit. So, there was a need to eliminate the incorrect postings to keep the sanctity of the data. We consider the latest posting as the correct one in case of duplicate postings.

Classification Models

We explored various supervised machine learning classifiers such as Random Forest (Liaw, Wiener et al. 2002), SVM (Suykens and Vandewalle 1999), and Logistic Regression (Hosmer Jr, Lemeshow, and Sturdivant 2013) as listed in Table 3. We evaluated each of these classifiers and used an ensemble of these classifiers to improve our results as different models will be better suited for different subsets of data.

After evaluating various conventional machine learning models, and ensembles, we realised we needed a hybrid classifier where different steps can be taken care by different engines. This was also done to imitate how the human agents complete the process and gave us insights about how we could model the solution for better results. Following this, we built a hybrid classifier as detailed in Figure 2. This decoupling of modules also enables us to change and improve

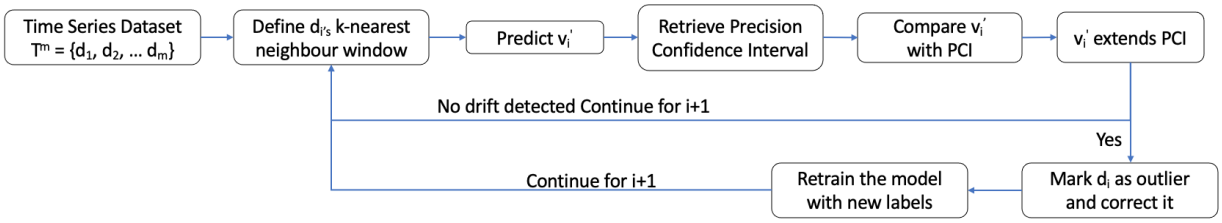

Figure 3: Algorithm for drift detection using sliding window precision

Item Description	Similar Description Retrieved
men tp whit system revit ld 125ml it	ment p prev compl ld 100ml it
airbag 120 x 225	airbag 225 x 120
tray 12x300g 16	tray green 12x9l (18cubes)
chocolate, milk, liquid, var.10 ra	chocolate drop milk 30% cocoa ra

Table 2: Example of semantically similar descriptions found retrieved by similarity engine

each independent segment to enhance the overall accuracy and efficiency.

The proposed solution was later tested on other countries as a litmus test for generalizability and scalability and to check how generic the solution is and the ease with which the solution can be applied for a different client or country.

Semantic Similarity Engine

The first step for tax code prediction using historical data is to find similar item descriptions from the historical data. In the current deployed system, it is done using an information retrieval (IR) system. All the descriptions from historical data are indexed. For a new description, this index is searched for the top-N (configurable parameter) best matching descriptions. This gives us the flexibility to search through all the descriptions in a time efficient manner and at the same time getting a ranked list of matching descriptions. The decoupling of different steps ensures that this engine can be later replaced by some other semantic similarity module if experiments prove that to be better for a different client, region, or language. The choice of using an IR system was also influenced by the fact that there is a huge imbalance in tax code labels, which would severely affect the performance of machine learning based classification models.

Some examples of item description and retrieved similar item descriptions are shown in Table 2. The item descriptions may contain quantity as numbers and other shorthand which may not be linguistically accurate.

Rule Engine

The tax code determination also depends on some well defined rules and criteria. For example, the tax code applicable on an item would be dependent on the country it is delivered to. These rules and criteria also vary with time. Hence, we have a modular rule engine where such rules can be configured easily. Once matching descriptions are found, we extract all the instances of these matching descriptions. These ranked descriptions are then filtered based on matching shipping addresses from the historic data. The shortlisted data points are further filtered for company codes and are checked for matching vendor details. The instances with exactly matching descriptions and the instances where the vendor details match, get a higher score. Since, this rule engine is also a standalone module in itself, the rules can be easily modified for a different setting.

Tackling Drift with Feedback

Model performances may deteriorate over time since the data distribution might change and labels for the same data points might also change. This is specially true in the case of tax code determination. Rules and regulations related to tax change over time. This results in *concept drift* (Widmer and Kubat 1996; Gama et al. 2004) where the properties of the label changes over time. For our use-case, this means that for the same set of data, the tax code might change over time. It becomes difficult to keep track of each and every change for a plethora of items, countries and clients. Similarly, for a particular client, the items they purchase, the vendors the items are purchased from and the geographically location of purchase and delivery can change. This would result in *data drift* (Quionero-Candela et al. 2009) which is the change in the distribution of data over time.

We follow a similar approach as detailed in Figure 3 to tackle both concept drift and data drift. When making a prediction using historical data, the proposed system gives more importance to the recent data points. Once the rule engine filters the most similar data points from the historical data, these data points are then sorted based on the recency of their posting. Then, out of the latest N data points, the most frequent tax code is selected. In case of a tie, the system then looks for the majority tax code across the whole filtered data points rather than the latest N. This helps us in tackling the drift in the data over time.

This approach also helps us flag any changes which have come up in the tax reporting because of regulations. For example, if the tax code changed for an item, the module initially might give a wrong prediction with high confidence. But after feedback from agents, the module would start decreasing the confidence of further predictions. The feedback would act as a deterrent and within $N/2$ predic-

tions, it would start predicting the correct tax code. For an instance, if $N = 5$, within 3 wrong predictions, the system would learn to predict the correct tax code. This way, the model also takes into account the inter-agent agreement for the feedback. A single feedback can not change the applicable tax code for a set of inputs. Only after a certain number of negative feedback, the predicted tax code would change thus reducing probable errors committed by agents. This design helps the system to tackle the concept drift which may arise in the system. In essence, this provides the system with a self-learning capability given the feedback.

Evaluation

Metrics

The metrics we consider are based on the fact that tax code determination invites audits and is governed by regulations. Wrong predictions can mean delayed payments, penalties, and audits. Considering these factors, we concentrate on the following metrics to evaluate the proposed system against other models:

1. **Accuracy** - The number of correct predictions made by the system out of all predictions.
2. **Precision** - The number of true positives among the total predictions. This helps because the data is highly imbalanced.
3. **Number of wrong predictions** - The proposed system predicts *"Unknown"* when the predicted confidence is below the defined minimum confidence. This metric helps us understand number of erroneous predictions out of all the predictions where a tax code was predicted. This helps in tracking the wrong results which may result in incorrect posting and penalties.
4. **Number of high confidence predictions** - High confidence predictions flow through the proposed system without any manual interference. The aim is to have as many predictions as possible in this category to save time and resources.
5. **Accuracy of high confidence predictions** - Since, high confidence predictions flow through without any manual validations by human agents, it becomes essential to have very high accuracy for high confidence predictions.

Experimental Protocol

We had received historical data with 35 fields for a lot of countries for a particular client. After some filtering, we were left with 14 fields. To understand the correlation amongst different fields, we ran different models with different set of features generated after pre-processing these fields. Eventually, the best results came when considering description, shipping addresses, vendor details, posting date, and company code. One of the key aspects is also the fact that a random split of data was not possible because time continuity was a factor. We split the data for train/test based on posting dates of invoices with a $80 : 20$ split. Subsequently, training data had 42733 data points and test data had 10745 data points.

We ran experiments with different models including linearSVC, logistic regression, and Random Forest. One reason for trying these set of classifiers was because we needed a score which could be treated as confidence. The classifier score is treated as a proxy for confidence. The proposed system was evaluated in two stages. Firstly without any online training or feedback i.e., without any changes to the learned models. Secondly, with online training where each test datapoint's correct label was used as a feedback for the system to simulate the actual online feedback scenario when the human agents would give feedback for predictions.

For our hybrid classifier, we experimented with various values of D_s and V_s. The best results in terms of high accuracy with a high number of high confidence predictions and needed precision recall was when $D_s = 0.5$ for exactly matching description, and $D_s = 0.35$ for fuzzy matches. Similarly, $V_s = 0.2$ if the vendor details matched. This was also the initial configuration that was deployed in production. Based on the learning from the agent feedback, the hyper-parameters for our deployed system were updated to $D_s = 0.55$ for exactly matching description and and $D_s = 0.5$ for fuzzy matches. Currently, the confidence threshold for minimum confidence(C_{min}) is set at 60% and maximum confidence(C_{max}) at 85%.

Training Results

The results obtained for 10745 test data points with a high confidence threshold of 75%, are detailed in Table 3. The proposed system with online training provided 92% accuracy and performed better than the other models in terms of overall accuracy, overall precision, and overall recall. It also outperforms other models in high confidence accuracy, precision, and recall by $2 - 5\%$. Although the number of high confidence predictions are higher for other models, they perform poorly on other metrics which may result in a lot of wrong postings and agents losing trust in the system over time. The other models also perform very poorly on the minority classes(tax codes). For the proposed system with online training the number of *Unknowns* also come out to be only 215.

Business Impact

The proposed system is deployed for a major multi-national company with annual accounts payable invoices volume of 5.5 million. Currently, the system is deployed for its invoices in Italy. At present, it has processed 22375 invoices with an accuracy of 94.46%. Also, the high confidence predictions ($\geq 85\%$) which auto-flow and does not require manual feedback has been more than 73%. The remaining invoices are sent for feedback to the agents. The accuracy achieved for high confidence is 99.54%. The number of *Unknowns* are 427 implying that the engine does not make a prediction for less than 0.2% data points. Also, 3181 data points were predicted with 100% confidence and all of them were correct predictions. The results are tabulated in Table 4.

The systems is being expanded to 19 more countries in Europe region. It is also in development for 19 countries in Latin America (LATAM) including countries such as Argentina, Columbia, Mexico. The scalable architecture of the

Model Characteristics	Precision	Recall	Accuracy	# of High Conf. Predictions	High Confidence Precision	High Confidence Recall	High Confidence Accuracy
Majority Class	0.65	0.81	0.80 (8658)	10745	0.65	0.81	0.8
Random Forest (Liaw, Wiener et al. 2002)	0.87	0.85	0.85 (9175)	10528	0.88	0.87	0.86
Linear SVC (Pedregosa et al. 2011a; Suykens and Vandewalle 1999)	0.87	0.86	0.86 (9265)	10212	0.89	0.89	0.88
Logistic Regression (Hosmer Jr, Lemeshow, and Sturdivant 2013)	0.87	0.88	0.87 (9412)	10351	0.89	0.9	0.89
Proposed system without online Training	0.89	0.83	0.84 (8740+327*)	7008	0.93	0.93	0.93
Proposed System with online Training	**0.93**	**0.92**	**0.92 (9743+215*)**	**8046**	**0.95**	**0.95**	**0.95**

Table 3: Experimental Results for our system and various models on test data. * refers to Unknown

Number of Invoices	Precision	Recall	Accuracy	Unknowns	# of HC. Predictions	HC Precision	HC Recall	HC Accuracy
22375	0.96	0.94	0.94 (21137/ 22375)	427	16589	0.99	0.99	0.99 (16527/ 16589)

Table 4: Results from the production system. HC refers to High Confidence.

proposed system enables real quick deployment into new environments and client accounts. Overall after the successful deployment for Italy, it is currently to be deployed in total of 85 countries for the same client. Its already in deployment for two more clients now that it has proven its edge over manual methodology.

For Italy, annual volume is ∼48,000 invoices where each invoice can have multiple items having different tax associated with them. On an average it takes 30 seconds for a human agent per invoice for determining the tax code. Even with a very conservative estimate of 1 item per invoice, with the implementation of tax prediction module, it amounts to saving of 400 work-hours per country per client annually. When it scales to other countries for this client, an exponential saving of work-hours is expected annually.

One of the claims and intentions of online learning via feedback was adjusting confidence thresholds as the engine classifies more data. This would help in auto-flowing of more invoices as well as keeping a high accuracy for high confidence predictions. Upon learning from the feedback received, we have adjusted the thresholds in the deployed system. In Figure 4, the number of correct and wrong predictions are plotted for the initial configuration of weightage given to different parameters and updated configuration.

Results for Other Countries

One of the major challenges for our system is the generalizability of the approach. The proposed system needs to be easily generalizable for other countries for the same client as well as other clients. The proposed system works well even for other countries in Europe region as evident by Figure 5. The proposed system shows high accuracy for all the 19 countries in Europe where the system is being deployed now. Also, the percentage of wrong predictions for high confidence predictions is also low across countries. Since we wanted to examine the generalizability of our approach, absolutely no changes were made to the configuration. The only change that was done was to train the model with the corresponding data. The metrics would further improve once the configuration of different weights for semantic similarity, rule engine and drift are updated based on the data specific to the countries. We hope that once it is deployed for other countries, the results would improve even further as the system learns from the feedback. This showcases that the proposed system can be easily deployed for other countries and clients with minimum or no effort, thus making it easily adoptable for reuse.

Deployment and Maintenance

The system is currently deployed as a single tenant service in a docker container. The service has four REST APIs for (i) */loading_relevant_data*: for uploading the data related to the tax mappings, (ii) */training*: to train the information retrieval system and rule based engine, (iii) */prediction*: to predict the tax code for a given item and its details, and (iv) */feedback*:

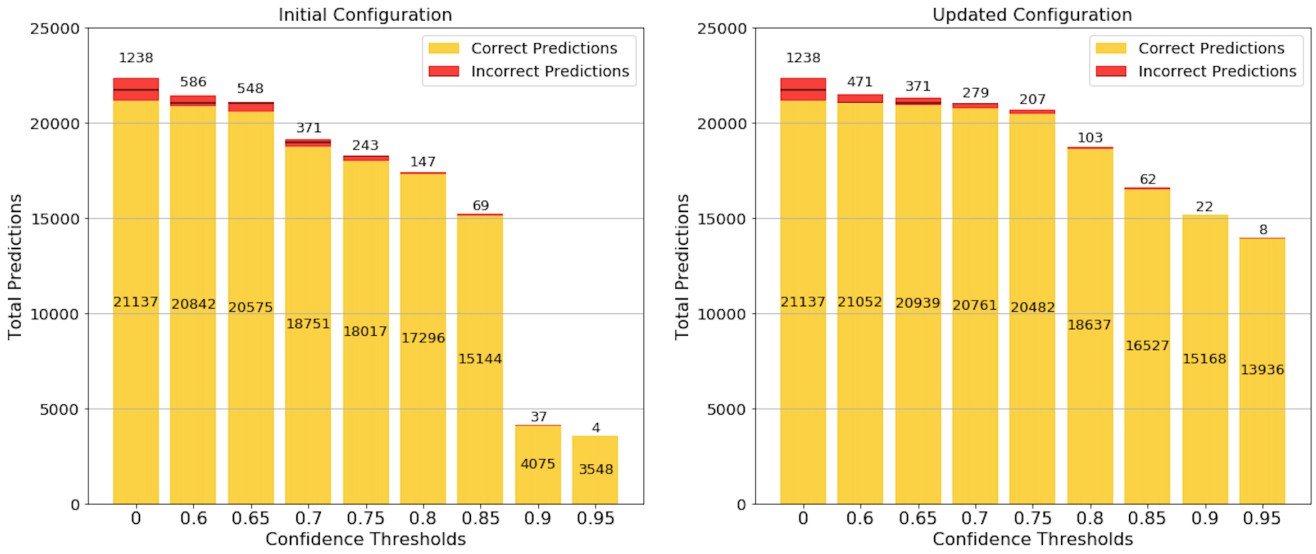

Figure 4: Number of correct and wrong predictions for initial and updated configurations

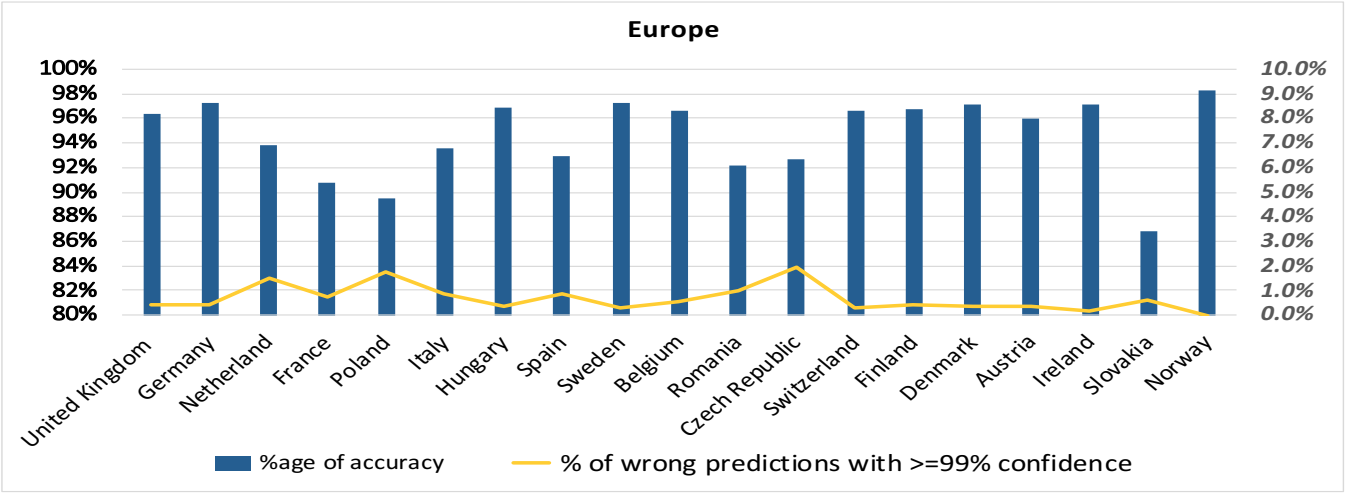

Figure 5: Results for other countries

for agents to give positive or negative feedback for a prediction. The entire development was done in python and APIs were created using python flask. Various machine learning models were trained using open source scikit-learn[1] library (Pedregosa et al. 2011b) . The IR system works on top of open source pylucene[2] library. The database used for storing historical and processing data is mongodb[3]. There is also a governance dashboard which keeps track of how the tax code determination system is performing, analysis of results and various data, and prediction distributions to understand the changes in the system.

The maintenance of the system includes adding new tax codes when a tax code changes or any other regulation changes, adjusting confidence thresholds as the system improves with the feedback, and building new rules if needed with changing business requirements. With respect to deployment for a new client and/or region, the only changes required are that the model needs to be trained on the region/client specific data and a one time configuration of different weights and confidence thresholds (8 parameters). Such a tunable system is essential as we cater to the needs to different customers in different production settings.

Related Work

The work by (Brady and Naughton 2009) discusses tax code determination using data stored in databases to determine the tax code. This would only work for exact data

[1]https://scikit-learn.org/stable/
[2]https://lucene.apache.org/pylucene/
[3]https://www.mongodb.com/

points appearing again. Also, this would not be able to handle data and concept drift easily. The work by (Maurya et al. 2020) describes matching semantic similarity of item descriptions between purchase order and invoice items in accounts payable. The difference being the matching is between limited number of items from 2 available documents. There is plethora of work related to data drift (Quionero-Candela et al. 2009) and concept drift (Widmer and Kubat 1996; Gama et al. 2004), but not something which tackles drift in the domain of tax code determination for accounts payable.

One of the other systems which handle tax code determination is the ERP system by SAP. But the automation stated in SAP's ERP system concentrates majorly at the purchase order level whereas our system is at the invoice processing level. The ultimate responsibility to update the correct tax code lies with invoice processing team. The tax code updated at PO only acts as a reference point during the invoice processing but the final and accurate tax code determination must be done during invoice processing. Hence, our solution determines the tax code at the invoice processing level considering both PO and invoice data. Additionally, we have considerable number of clients still using ERP systems other than SAP's ERP system. Our proposed system can integrate to any ERP system and not only SAP's ERP system.

Conclusion and Future Work

We have developed a configurable system aimed at automating tax code determination for different product and services. The underlying hybrid classifier looks at various features such as item description, vendor details, origin country, and destination country and predicts the most appropriate tax code for a particular item. The system also employs online training via feedback from agents thus reinforcing the model with possible changes or drifts in the data distribution over time. The feedback also helps us to fine-tune the performance thus pushing more and more items directly for posting over time. These features help the system in becoming self-reliant over time. Because when tax code changes, it is important for system to start curating the data for new tax label. Hence feedback provides a way to capture that. Also, since each part of the system is a stand-alone module, it can be easily replaced based on changing data characteristics or client requirements or in lieu of a better performing module. The experiments done with historical data from other 19 countries highlight the generalizibility of the proposed system.

The data we collect can help us to analyse performance of different agents as well as vendors. Also, when a tax code or category changes due to changes in tax regime, the system would need some time and data points from feedback before it starts predicting changed tax code. As a part of future extension, we need to come up with a solution where the old data can be readjusted depending on the changing tax regime.

References

Agha, A.; and Haughton, J. 1996. Designing VAT systems: Some efficiency considerations. *The Review of Economics and Statistics* 303–308.

Anderson, S. P.; De Palma, A.; and Kreider, B. 2001. The efficiency of indirect taxes under imperfect competition. *Journal of Public Economics* 81(2): 231–251.

Brady, K. P.; and Naughton, S. T. 2009. Information processing system for determining tax information. US Patent 7,627,504.

Decoster, A.; Loughrey, J.; O'Donoghue, C.; and Verwerft, D. 2010. How regressive are indirect taxes? A microsimulation analysis for five European countries. *Journal of Policy analysis and Management* 29(2): 326–350.

Devereux, M. 2016. Measuring corporation tax uncertainty across countries: Evidence from a cross-country survey .

Gama, J.; Medas, P.; Castillo, G.; and Rodrigues, P. 2004. Learning with drift detection. In *Brazilian symposium on artificial intelligence*, 286–295. Springer.

Hoff, K. 1991. Land Taxes, Output Taxes, and Sharecropping Was Henry George Right? *The World Bank Economic Review* 5(1): 93–111.

Hoppe, T.; Schanz, D.; Sturm, S.; and Sureth-Sloane, C. 2018. What are the Drivers of Tax Complexity for MNCs? Global Evidence. *Intertax* 46(8/9): 654–675.

Hoppe, T.; Schanz, D.; Sturm, S.; and Sureth-Sloane, C. 2019. Measuring tax complexity across countries: A survey study on MNCs .

Hosmer Jr, D. W.; Lemeshow, S.; and Sturdivant, R. X. 2013. *Applied logistic regression*, volume 398. John Wiley & Sons.

Ingraham, L. R.; and Karlinsky, S. 2005. Tax professionals' perceptions of small business tax law complexity. *Tax Notes* 107(1): 79.

Keen, M.; and Lockwood, B. 2010. The value added tax: Its causes and consequences. *Journal of Development Economics* 92(2): 138–151.

Keen, M.; and Smith, S. 2006. VAT fraud and evasion: What do we know and what can be done? *National Tax Journal* 861–887.

Khurana, A.; and Sharma, A. 2016. Goods and services tax in India-A positive reform for indirect tax system. *International Journal of Advanced Research* 4(3): 500–505.

Liaw, A.; Wiener, M.; et al. 2002. Classification and regression by randomForest. *R news* 2(3): 18–22.

Little, I. M. 1951. Direct versus indirect taxes. *The Economic Journal* 61(243): 577–584.

Maurya, C. K.; Gantayat, N.; Dechu, S.; and Horvath, T. 2020. Online Similarity Learning with Feedback for Invoice Line Item Matching. *arXiv preprint arXiv:2001.00288* .

McKerchar, M. 2005. The impact of income tax complexity of practitioners in Australia. *Austl. Tax F.* 20: 529.

Newbery, D. M. 1986. On the desirability of input taxes. *Economics Letters* 20(3): 267–270.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011a. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12: 2825–2830.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. 2011b. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12: 2825–2830.

Quionero-Candela, J.; Sugiyama, M.; Schwaighofer, A.; and Lawrence, N. D. 2009. *Dataset shift in machine learning*. The MIT Press.

Schaeffer, M. S. 2002. *Essentials of accounts payable*. John Wiley & Sons.

Schenk, A.; Thuronyi, V.; and Cui, W. 2015. *Value added tax*. Cambridge University Press.

Spengel, C.; Evers, L.; and Zinn, B. 2012. *Unternehmensbesteuerung in Deutschland: Eine kritische Bewertung und Handlungsempfehlungen für die aktuelle Steuerpolitik*. ZEW Gutachten/Forschungsberichte.

Suykens, J. A.; and Vandewalle, J. 1999. Least squares support vector machine classifiers. *Neural processing letters* 9(3): 293–300.

Tait, M. A. A. 1988. *Value added tax: International practice and problems*. International Monetary Fund.

Widmer, G.; and Kubat, M. 1996. Learning in the presence of concept drift and hidden contexts. *Machine learning* 23(1): 69–101.