This CVPR 2020 workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Information Extraction from Document Images via FCA based Template Detection and Knowledge Graph Rule Induction

Mouli Rastogi[1], Syed Afshan Ali[1], Mrinal Rawat[1], Lovekesh Vig[1], Puneet Agarwal[1], Gautam Shroff[1], and Ashwin Srinivasan[2]

[1]TCS Research, New Delhi
[2]BITS Pilani, Goa Campus
{mouli.r, afshan.syed, rawat.mrinal, lovekesh.vig, puneet.a, gautam.shroff}@tcs.com, ashwin@goa.bits-pilani.ac.in

Abstract

We view information extraction from document images as a complex problem that requires a combination of 1) state of the art deep learning vision models for detection of entities and primitive relations, 2) symbolic background knowledge that expresses prior information of spatial and semantic relationships, using the entities and primitive relations from the neural detectors, and 3) learning of symbolic extraction rules using one, or few examples of annotated document images. Several challenges arise in ensuring that this neuro-symbolic software stack works together seamlessly. These include vision-based challenges to ensure that the documents are "seen" at the appropriate level of detail to detect entities; symbolic representation challenges in identifying primitive relations between the entities identified by the vision system; learning-based challenges of identifying the appropriate level of symbolic abstraction for the retrieval rules, the need to identify background knowledge that is relevant to the documents being analyzed, and learning general symbolic rules in data-deficient domains. In this paper, we describe how we meet some of these challenges in the design of our document-reading platform. In particular, we focus on use cases with multiple templates which additionally involves finding structurally similar images in large heterogeneous document image collections. An adaptive lattice based template allocation module was utilized for evaluating document similarity based on both textual content and document structure. A knowledge graph is used for capturing document structure and a relational rule learning system is employed on the knowledge graph for generating extraction rules. Experiments on a publicly shared data-set [1] of 1400 trade finance documents demonstrates the viability of the proposed system.

1. Introduction

With the widespread usage of mobile cameras and fast scanners to capture document images, the problem of downstream information extraction from such documents has become more acute. Despite the recent strides that deep learning has made in computer vision and natural language understanding, a generalized solution to the problem of information extraction from complex document images remains elusive due to a number of challenges. Humans perceive documents using a combination of visual and textual cues, many of which are not captured by most modern OCR engines. Additionally, humans also draw on significant domain specific and universal (common sense) background knowledge to relate and reason over a document's visual and textual components. A system for information extraction from real world document images must incorporate these elements in order to match human performance on information extraction. In this paper, we outline the design of our document extraction platform, that incorporates the above components to fully utilize the visual pattern recognition capabilities of deep learning, the relational representation power of knowledge graphs, and the deductive reasoning capabilities of symbolic learners for learning extraction rules and incorporation of background knowledge.

In particular, we focus on real world use cases that involve large volumes of documents belonging to a (potentially unknown) set of templates, where a template defines an invariant spatial and semantic structure. As documents are introduced into the system, a knowledge graph with a

[1]https://drive.google.com/open?id=1BjysqlaKRylwCi3RaesnvAjqqvWM8GWG

fixed schema based on background knowledge is used to capture the spatial and semantic relationships present in the document. A lattice based similarity metric utilizes both the spatial and semantic information in the document to determine if the document structure adheres to any of the apriori known templates. If a new document structure is detected then a fresh set of symbolic rules (corresponding to paths in the knowledge graph) are learnt for information extraction from documents belonging to the new template. Any future documents belonging to the same template are automatically processed via the learnt rules. Note that the system only requires a single annotated document per template, as it employs a powerful one-shot symbolic rule learner.

The primary contributions of the paper are as follows:

1. The paper presents the architecture for an end-to-end system for information extraction from streaming document images.

2. We identify the key vision problems and propose potential solutions relevant for document understanding.

3. A novel technique for one-shot rule induction from knowledge graphs is proposed, capable of learning rules for multi-line entities.

4. We propose a document classification technique based on Formal Concept Analysis (FCA), that imbibes both spatial and semantic content to allocate documents to prior/new templates.

5. The system is tested on a dataset of 1400 scanned documents from seven different templates. This dataset is made available to the community for future research.

The rest of the paper is organized as follows: Related work on document understanding, template detection and learning from knowledge graphs is presented in Section 2. The problem definition and solution pipeline are presented in Section 3 along with a formal definition of the problem. In Section 4, we demonstrate our experiments and present the results of our experimental evaluation, before offering conclusions in Section 5.

2. Related Work

A pre-requisite for information extraction is document template identification, a problem for which several techniques have been proposed. Breuel[2] developed a matching technique that was able to capture the layout of complex documents by generating a hierarchical representation using XYtrees and tree edit distance to identify the correct template for a test document. Hamza[8] utilized a bottom-up approach involving keyword extraction and clustering to generate high level structures. The document is represented as a graph with these high level structures as nodes and edges indicative of spatial relationships. Schultz *et al*. [17] developed an invoice reading system, with documents represented by a graph and nodes split into key-nodes and data-nodes, corresponding to keywords and values (numbers, dates etc). Graph probing is used to find the correct template. Our work builds upon these prior approaches and the recent work in Formal Concept Analysis (FCA) [7] which represents the document as a lattice for not only existing template identification but also for detecting novel templates for streaming documents in real time. This representation model is more advanced since except for the content, i.e., words in the document, also considers the layout information, i.e, their location in the document.

Rule learning over knowledge graphs is a well studied field [5, 6], however most approaches attempt to mine a knowledge graph for filling in missing facts and inferring all possible rules, using logic based techniques[22] as well as graph embeddings[9]. The objective here is to learn entity specific rules that would yield a specified target node corresponding to a desired entity value. To that end we propose a novel rule induction algorithm in Section 3.3 capable of learning extraction rules with a single annotated sample document. End to end document processing engines have previously been proposed [13, 14]. However, these systems are largely geared towards the key-value extractions from standard forms based on relatively simple spatial rules. Our system is more versatile, can adapt to multi-line entity values, utilizes conjunctions of complex multi-hop subgraphs with semantic relationships, and is robust to noise.

3. Our approach

Consider a streaming set of documents $d_1, d_2, ...$, entering the system where each document belongs to one of N different templates $T_1, T_2, ..., T_N$. The system objective is to extract values for a predefined set of entities $e_1, e_2, ..., e_n$ (such as invoice number, date or billing address) from each document. Templates are not known in advance and manual annotation is requested when the system encounters a document from a previously unseen template. [2]

On receipt of a document the system proceeds to run the document though a series of vision APIs detailed in Section 3.1 in order to extract information about different visual elements such as text, boxes, tables and charts. This extracted information is then used to populate a knowledge graph with a predefined schema that captures the spatial and semantic relationships between the detected elements. The relationship mapping schema and the resulting knowledge graph is detailed in Section 3.2. The encoded relationships are then used to determine whether the document belongs to a new or pre-existing template via a lattice based technique

[2]While multiple annotations per template may be permissible for practical deployment, for the current system we only permit one annotated document per template.

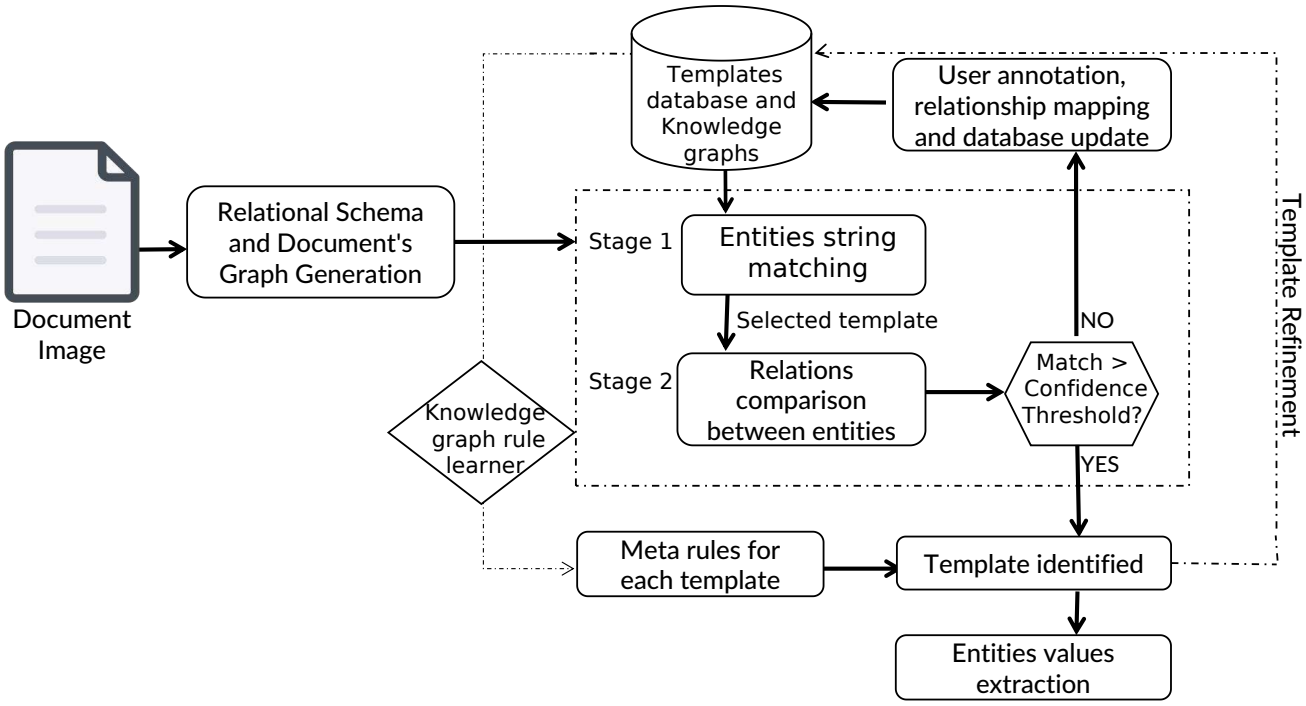

Figure 1. Full System Architecture

rooted in Formal Concept Analysis[7] described in Section 3.4.2. If a new template is detected, manual annotations are provided and symbolic rules are learnt on the knowledge graph, as outlined in Section 3.3. The new template and the associated learnt rules are then stored in the system for application to any future documents adhering to the same template. The full system architecture is depicted in Figure 1.

3.1. Document Vision APIs

Prior to analyzing the contents of the document, there are significant vision based challenges that modern document processing engines tend to ignore, here we outline how we address these challenges:

Document De-noising Real world documents are often noisy, with scanned documents often suffering from blur effects, faded text, watermarks, scanning artifacts, and wrinkles. The noise in these documents often leads to downstream OCR/ICR and other errors. Traditional methods for denoising images[4, 1, 18] are often too specialized and fail for novel noise distributions across different document datasets. Training a typical deep model for denoising documents requires significant volumes of paired clean and corresponding noisy document images. Our system gets around this problem by utilizing a cycle GAN as decribed in [20] to learn the mapping of the distribution of noisy documents to that of clean documents.

Visual Cues: Humans utilize several vision based cues such as font styles, lines and text structure while reading documents, these are often ignored by many current systems which focus purely on text extraction. We use a combination of deep learning and traditional vision to extract this additional information using the vision APIs described in [15].

Text Detection/Recognition: Despite recent advances, text detection and recognition accuracies [21, 3], particularly for handwritten text are not yet comparable to human accuracies and are extremely sensitive to noise, orientation and scale. The problem becomes even more acute for multilingual documents where training data is sparse. Our system utilizes the state of the art deep models for handwritten text recognition and detection, in addition to models for language detection.

Table Detection and Tabular Structure Identification: Often real world invoices and other documents contain data in tables with highly variable structure, and while deep learning has made important strides in table detection [12, 16], extraction techniques that generalize well across unseen table formats remains a challenge. The proposed system uses a combination of deep models and common knowledge about tables along the lines of [12].

Information Extraction from Charts: Business docu-

ments often contain Histograms, Graphs, and other complex data visualizations that capture important information. Recent methods using MAC networks to reason over charts have made progress[19] and we utilize a similar approach in this paper to process visual charts.

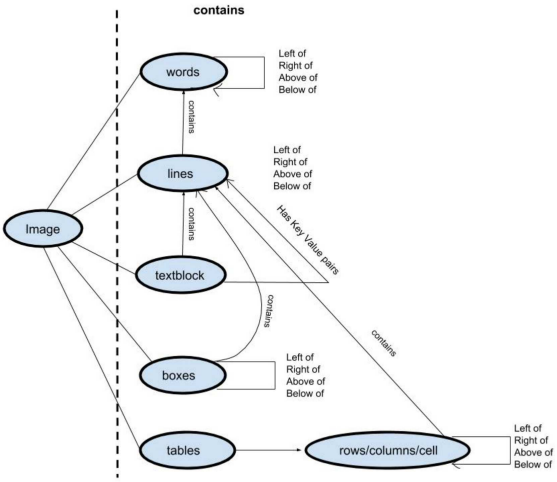

Figure 2. DeepReader Schema

3.2. Relationship Mapping

Post detection, the semantic and spatial relationships between the different visual and textual elements have to be computed. This includes grouping text into blocks, unravelling complex tabular structures, extracting, inferring relationships from charts and graphs, and grouping complex visual structures like checkboxes and form field boxes. Effective relationship mapping requires incorporation of background knowledge for understanding the consistent structure of textblocks, tables and checkboxes, and correcting for OCR errors. Background knowledge and natural language understanding is also useful for semantic data typing for commonly occurring patterns like dates, city names and addresses. Additional domain specific knowledge may also be required in certain specialized use-cases. These complex relationships are stored in a knowledge graph with a schema as shown in Figure 2. The nodes in the knowledge graph are typed and represent different visual entities such as words, textlines, blocks and tables. These nodes may be suitably enriched with additional attributes to incorporate domain knowledge such as semantic data types.

3.3. Knowledge Graph Rule Induction

For our application, we are interested in rules for extracting textual content (node) corresponding to an entity value from a particular document template, in contrast to other rule learners that mine the KG for frequently occurring patterns [9]. We refer to the node corresponding to the target entity value as the target node N_t, and leverage the fact that every subgraph connected to the target node represents a potential rule r for extraction as shown in Figure 3. To prevent a combinatorial search we employ a depth bound d and a maximal branching factor b for any node while searching for rule subgraphs. The set of all such rules is denoted by R as shown in Figure 4. However, each such rule $r \in R$ found may (i) be overly specific to the current document or (ii)lead to spurious extractions, i.e. be overly general.

In order to address the latter, we enforce the closed world assumption and eliminate rules R_s that yield subgraphs connected to any target nodes other than t. To address the former we generate a noisy clone of the original KG' wherein the entity values are deliberately altered by adding textual noise. Any rules that fail on the noisy clone KG' are pruned, we refer to these rules as R_g. The final set of learnt rules is then $R - R_s - R_g$.

S.No.	Conjunction Level	Rule
1	1	"USD"(check text) – "wIsLeft" – groundtruth
2	1	"Value Presented For USD 720"(check text) – "has"(relationship) – "For"(check text) – "wIsLeft" – USD(dont check text) – "wIsLeft" – groundtruth
3	2	Line "Value Presented For USD 720"(check text) – "has"(relationship) – "USD" – "wIsLeft"(relationship) – groundtruth AND word "Value"(check text) – "in"(relationship) – "Value Presented For USD 720" – "has"(relationship) – groundtruth

Figure 4. Rules generated for the Knowledge Graph in Figure 3

3.3.1 Learning Meta-Rules

Occasionally entity values span a variable number of textual elements, i.e. may comprise of multiple words, lines or blocks. Such entity values correspond to multiple target nodes in the KG. Learning rules for multiple-target nodes involves learning a meta-rule which comprises of a starting target node n_s, an ending target node n_e, and a relationship subgraph R_G. Rules for extraction of n_s and n_e are induced as in Algorithm 1. The intermediate nodes are extracted via repeated traversal of the knowledge graph from the starting node until the ending node via the relationship subgraph as shown in Figure 3. For our application a linear subgraph (or path) is considered but in principle any subgraph that connects two nodes may be employed. At test time, the rule for the starting node is applied, followed by repeated application of the meta-rule (from the relationship subgraph) until either the ending node rule is found (i.e. its extraction rules are satisfied) or a maximum depth is attained indicating a rule failure.

3.4. Template Detection

Our framework considers a set of scanned documents like invoices, bills, or receipts wherein each documents can come from one of several different unseen templates. A template defines the document structure and all documents that belong to the template must adhere to the same

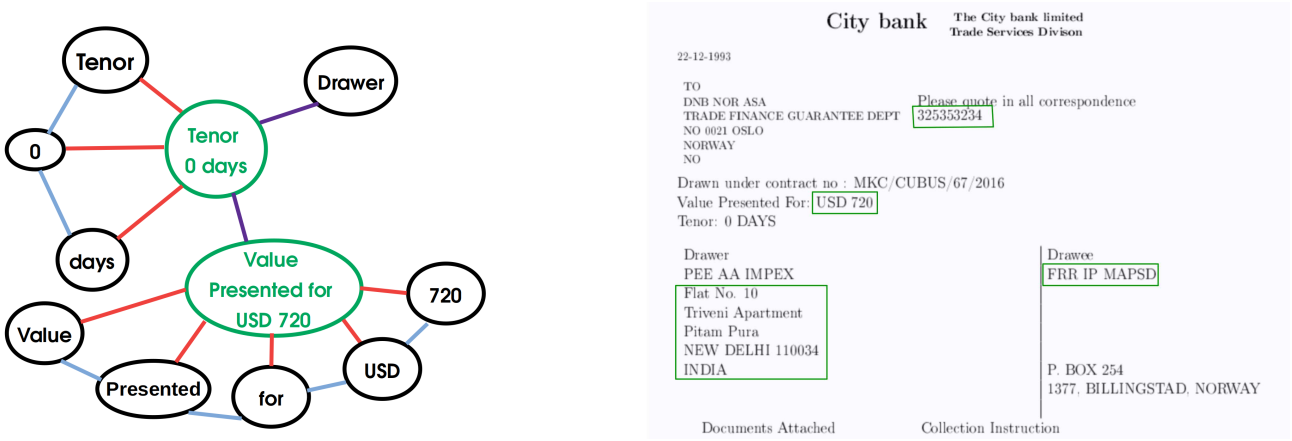

Figure 3. Knowledge Graph(Left) and corresponding annotated document (Right), the different colored edges and nodes represent different relation types (above-below, left-right) and entity(words, lines) types respectively. Each subgraph connected to an entity(node) can be converted into a potential extraction rule for that node. Note that sometimes the extracted value may lie across multiple nodes (lines), for instance the address field, and this would require a meta-rule

structure. More formally, given a document template with static components S and dynamic components C, the spatial/semantic relationships between static components of all documents from the same template must hold consistently.

Our aim is to classify a series of scanned documents to their respective templates, and also to identify newly encountered templates for manual annotation. The template detection module uses both the textual content as well as the relations of the entities(lines) with other entities present in the documents. The identification of relevant templates for each document is performed via a similarity(confidence) metric.

Formally, a Template **t** can be described as **t** = (line_1, above_1, left_1,..., w_1), $\cdots$, (line_n, above_n , left_n,... w_n) where $line_i$ indicates an entity (both static and dynamic) in the template, $above_i$, $left_i$ describes its spatial relations with other entities in the template and w_i denotes a weight value representing the confidence that $line_i$ will be found in a document **d** of template **t** as shown in Figure 5.

Next, we compute the similarity of each template in the template database with the incoming document based on the number of matching entities. This yields the template which has highest textual similarity with the document. The chosen template is then used in the next stage to reduce the document graph for checking its structural similarity with the templates.

To check for its structural similarity, a second level match is required that consists of the following steps:

3.4.1 Reduction of Document Graph:

This step, firstly replaces the dynamic entities present in the document with a token <**val**>, which are identified from the chosen template. Furthermore, as there are often digitization errors in the entity names, so we use fuzzy matching (based on Levenshtein distance [23]) to detect dynamic entities from the chosen template. The next key process in this step is to compare the paths of each pair of distinct nodes in the template's graph with corresponding nodes in the document's graph. The execution of this process is as follows:

- Pick any two distinct nodes(entities) from the chosen template graph and find all nodes(entities) between them in the document graph.
- Remove all the in between nodes from the document graph and create a direct relationship between them to get the shortest path. For example, if the relationship between all the intermediate nodes was 'ABOVE', the new relationship also created as 'ABOVE'.
- Repeat steps 1 and 2 for all pair of distinct nodes in the template graph.

These steps results in a reduced document graph which is further used to detect layout similarity (see Figure 5).

3.4.2 Lattice based Structural Similarity:

The reduced document graph computed in the previous step is then checked for the layout similarity with each of the templates in the template database. If match of the document is below the confidence threshold with the chosen template, then the document is sent for manual annotation and is identified as a new template in the template database. This template layout matching procedure, is inspired by a lattice based approach using Formal Concept Analysis.

Formal Concept Analysis:

Formal Concept Analysis (FCA) is a mathematical theory

Algorithm 1 Inductive Rule Learning

Input : A knowledge graph KG for document D, a set of m attributes $A = (a_1, a_2..., a_m)$ for each node n_i, a set of binary operators O defined over elements of A, a target node n_t corresponding to the entity value to be extracted, a node branching factor b, depth bound d

Output : Rule set R for extracting node N_t

- With n_t as the root, traverse the KG to obtain the set of all possible subgraphs S_G rooted at n_t, with branching factor $<= b$ at every node and maximum path length for any node from $n_t <= d$
- For each such subgraph s_G,
  1. For every node pair n_i and n_j in s_G:
     (a) If n_i and n_j connected by a relation R in s_G, generate a clause to that effect i.e. $R(n_i, n_j)$
     (b) Choose a binary operator $op \in O$ and a subset of attributes $s_A \subset A$ on which the conditional operator holds for nodes n_i, n_j, i.e. $op(a_{ik}, a_{jk})$ must be true $\forall k$ s.t. $a_k \in s_A$, where a_{pm} refers to the value of attribute a_m for node p.
     (c) Generate clauses for each such pair of corresponding attributes in n_i and n_j. i.e. $attr(n_i, k, a_ik), attr(n_j, k, a_jk), op_l(a_{ik}, a_{jk})$
  2. Create a rule whose body is the conjunction of all the clauses discovered for all node pairs, and whose head is simply the target node i.e. $entity\_name(D, n_t)$, note that this includes node pairs involving n_t. Add this rule to the rule set R.
  3. Repeat Steps 1-2 with all possible choices of operator-node pair combinations until no new rule can be generated for s_G
- Apply rules in R to the KG and eliminate over general rules that yield non-target nodes $n \notin n_t$
- Create a noisy clone KG' of the original knowledge graph by adding noise to all the dynamic entity value nodes
- Eliminate over specific rules from R that do not yield the target nodes in KG'

of concept hierarchies based on Lattice Theory [10]. Data is represented as a two-dimensional context of objects and attributes. Traditionally, a concept is determined by its extent and its intent. The extent of a concept consists of all objects that have a set of common attributes in a context, while the intent consists of all attributes that are considered valid for that context. The hierarchy of concepts is given by the relation of a subconcept w.r.t. a certain superconcept, i.e., the extent of a subconcept is a subset of the extent of its superconcepts, while the intent of a superconcept is a subset of the intent of its subconcepts.

We use a lattice based approach by viewing a given document and templates as formal concepts representing a triplet of the form **(O, P, I)** where **O** is the set of entities present in the document, **P** is the set of spatial relations corresponding to each entity and **I** is a binary relation between **O** and **P**. Each triplet produces a lattice structure for the corresponding document. The lattice structure consists of a set of formal concepts of the form **(A,B)**, where $\mathbf{A} \subseteq \mathbf{O}$, $\mathbf{B} \subseteq \mathbf{P}$, $\mathbf{A'}=\mathbf{B}$, and $\mathbf{B'}=\mathbf{A}$. The spatial relationships of each entity in a document are represented by the intents of the lattice constructed for that document.

Using the approach of lattices, we extract the similarity relationships with the help of knowledge graphs, and use it to compare the structure of new document with each of the template in the template database. As a first step, we extract the relationships for every entity in the reduced graph of document. The relationships for every node is stored in the form of a tuple. For example:

$$node_i : (node_j(ABOVE), node_k(LEFT))$$

The tuples generated correspond to the intents generated from FCA. Next, we generate the relationship tuples for each of the templates. The tuples of the reduced graph of the document are then compared with the tuples of each template in the template database to select a template, **t** which has the highest match factor(confidence), $\mathbf{c}_t$ out of all the templates where $\mathbf{c}_t$ is above the set threshold value, γ. If $\mathbf{c}_t$ is below the value of γ for all the templates in the template database, the document is identified as a new type of document and routed for manual annotation.

3.4.3 Template Refinement

As documents stream into the system, new templates are identified and the template database is continuously updated. If a match to an existing template is found, the template entity weights used for producing the similarity matching score are increased or decreased depending on their presence or absence in a test document. Thus a template undergoes continuous refinement as data for that template streams into the system.

Mechanism:

Weights of all entities is initially set to 1. If a template $\mathbf{t} \in \mathbf{T}$ is predicted for an incoming document **d**, then the template **t** is updated based on **d** according to the procedure:

For each entity, **e** in the template **t**, our method finds its

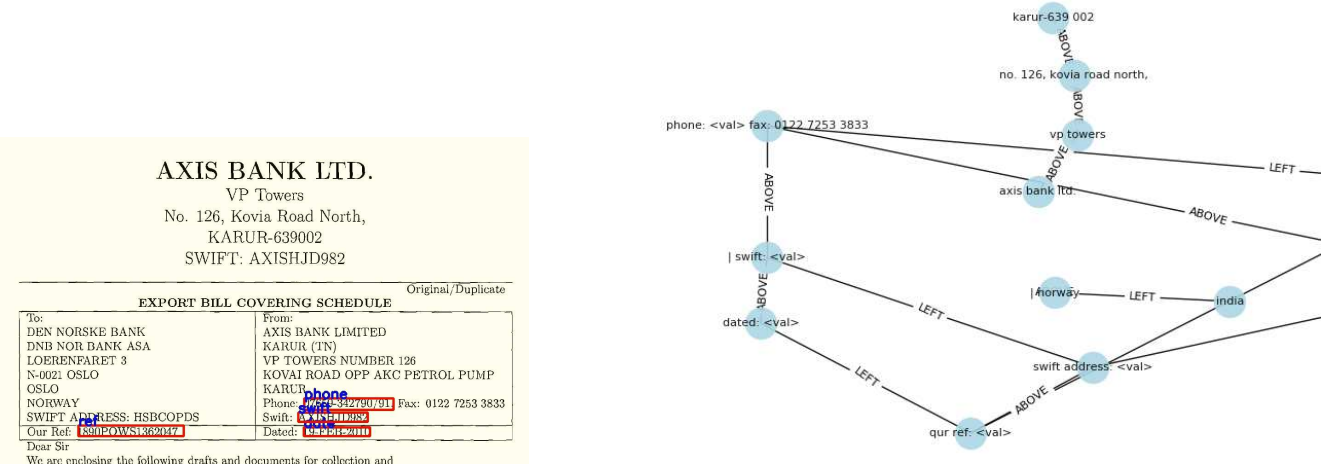

Figure 5. Annotated Document(Left) and corresponding Template Graph(Right). The lines of document in red boxes represents the annotated entities in the document, where each of them represents a dynamic field present in the document. Each node and edge of the template graph represents an entity and its spatial relations with other entities respectively.

matching entity **e1** in the document **d**. Weight of entity **e** is refined based on the corresponding **e1** entry. If **e** does not appear in **d**, **w** is decreased by a penalty factor **p** and after the weight **w** falls below the set threshold value θ , that entity is removed from the template as it consists of customer particular information. Iterating this through a certain number of documents, the template file results into containing the static entities only that are present in majority of the documents classified as that template.

3.5. Information extraction

Once a template has been associated with an incoming streamed document, the learnt extraction rules for that template are applied to the documents knowledge graph. However, due to noise in the scanning process or OCR errors some of the rules will fail or yield incorrect values (target nodes). A majority vote over the rule outputs would likely still yield the correct value, as rules that succeed in extracting the correct value would agree more often than rules that do not. The resulting majority vote value is then subject to domain specific validation rules (such as the amount entity has to be a real valued number) prior to output.

4. Experiments and Results

We used a publicly shared dataset of 1400 scanned bank trade finance documents to evaluate our method. The dataset consists of documents of seven different templates i.e. Axis, Citi, CMB, DBS, DutchBangla, Hangseng and Shanghai with 200 documents from each template. Documents from different templates are introduced at random into the system, and the system dynamically identifies novel templates and generates extraction rules for them after manual annotation of a single document for each template. We demonstrate the viability of our system for both template detection and information extraction.

S.No.	Template	Precision	Recall
1	Axis	1.00	1.00
2	Citi	1.00	1.00
3	CMB	1.00	0.74
4	DBS	1.00	0.98
5	DutchBangla	1.00	0.77
6	Hangseng	1.00	1.00
7	Shanghai	1.00	0.99
	Overall	**1.00**	**0.93**

Table 1. Precision and Recall of template classifier for each template.

4.1. Template Detection

We set a similarity threshold (confidence threshold),γ as *0.6* for identifying novel templates. For a given incoming document, **d**, the template having the highest confidence value above the threshold, would be assigned to **d**.

The similarity threshold was set so we are able to achieve high precision while compromising on recall. This implies that there are no false classifications made by our classifier whereas occasionally a noisy document may get misclassified as belonging to a novel template. However, each such document is simply routed for manual annotation to (i) build on a new template structure or (ii) re-assigned to an existing template.

Furthermore, for the template refinement module we set the penalty factor, p as *0.05* for a non-matching entity and a threshold value, θ as *0.5* which makes the template more robust with each new classified document by removing dynamic entities. This step shows gradual yet signif-

S.No.	Entity	Axis	Citi	CMB	DBS	Dutch Bangla	Hangseng	Shanghai	Mean Accuracy
1	Acc number	98.5	96.5	-	-	-	-	-	**97.5**
2	Amount	98.5	90.5	91.0	100.0	98.5	99.5	92.7	**95.8**
3	Date	100.0	94.5	98.5	77.5	98.5	94.5	95.5	**94.1**
4	Phone	97.5	-	100.0	94.0	-	-	89.3	**95.2**
5	Ref	91.5	-	98.5	89.5	98.5	100.0	87.6	**94.3**
6	Swift	78.0	78.0	84.5	91.5	98.5	100.0	92.1	**88.9**
7	Tenor	98.0	100.0	99.5	100.0	98.5	100.0	100.0	**99.4**
8	Drawee	83.0	90.0	70.5	95.0	98.5	89.5	91.5	**88.3**
9	Drawer	93.5	78.5	74.5	88.5	98.5	96.0	87.6	**88.2**
	Overall								**93.5**

Table 2. Accuracy(%) of different entities over 200 documents for each template along with their respective mean values.

icant improvement in the confidence value of the incoming documents that results in an increase in classification ACCURACY from 92.36% to 92.64% .

The overall result for this module provides a PRECISION = 1.00 and RECALL = 0.93 thereby giving F1 SCORE = 0.96 and ACCURACY = 92.64%. The precision and recall values for each template is provided in Table 1.

4.2. Information Extraction

To evaluate the quality of our results for this module, we compare the system's output values, v_{i1},...,v_{ip} for each predefined entity, e_i with their corresponding ground truth values, g_{i1},...,g_{iq} for documents belonging to one of $m = 7$ templates T_k, *k = 1,...,m* as a measure of accuracy, i.e. for an entity e_i of a template T_k,

$$ACCURACY = \frac{No.\ of\ documents\ s.t.\ v_{ir} = g_{is}}{Total\ no.\ of\ documents\ for\ T_k},$$

$$where\ r = 1, ..., p\ and\ s = 1, ..., q$$

checks for an exact match between the predicted value and the ground-truth. We also obtain the accuracy of our rule learner disregarding OCR errors in the final entity string value. In our implementation, we used the Jellyfish library[11] for this which helps in approximate and phonetic matching of strings. The ACCURACY for each predefined entity from all the seven different templates with OCR errors is showcased in Table 2, which results in overall ACCURACY of 93.5%. Without OCR errors the overall ACCURACY for obtaining the correct target node (entity value) was 100% which demonstrates the robustness of our rule induction system.

Thus, all inclusive out of 1400 documents of trade finance from different banks, 1297 documents were classified correctly, while the information extraction module was able to obtain the correct node or textual entity 100% of the time. However, due to OCR errors with the Tesseract OCR package Table 2 shows some decline in the end-to-end accuracy for each entity, but the numbers remain high.

5. Conclusion

This paper presents the detailed architecture and implementation for a templatized document information extraction engine. The paper outlines different steps involved in unravelling the visual and semantic structure of a document. These include developing solutions to the numerous vision problems during extracting of visual elements from the document, inferring spatial/semantic relationships between the different elements and learning rules for automated extraction. We introduce a novel technique for document classification based on Formal Concept Analysis that allows streaming documents to be allocated to a known template or sent for manual annotation. Additionally we provide algorithms for rule induction and meta-rule learning from knowledge graphs for single and multi-line entity extraction. Results are encouraging and demonstrate that the system is able to perform extraction with a very high degree of accuracy, significantly reducing manual effort. In the future we wish to extend this system to documents that do not adhere to any template, and where subsets of entity extraction rules may apply to different documents.

References

[1] Mudit Agrawal and David Doermann. Clutter noise removal in binary document images. In *2009 10th International Conference on Document Analysis and Recognition*, pages 556–560. IEEE, 2009.

[2] Thomas Breuel. A practical, globally optimal algorithm for geometric matching under uncertainty. *Electronic Notes in Theoretical Computer Science*, 46:188–202, 08 2001.

[3] Arindam Chowdhury and Lovekesh Vig. An efficient end-to-end neural model for handwritten text recognition. In *British Machine Vision Conference 2018, BMVC 2018, Northumbria University, Newcastle, UK, September 3-6, 2018*, page 202, 2018.

[4] Kuo-Chin Fan, Yuan-Kai Wang, and Tsann-Ran Lay. Marginal noise removal of document images. *Pattern Recognition*, 35(11):2593–2611, 2002.

[5] Mohamed H Gad-Elrab, Daria Stepanova, Jacopo Urbani, and Gerhard Weikum. Exception-enriched rule learning from knowledge graphs. In *International Semantic Web Conference*, pages 234–251. Springer, 2016.

[6] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. Fast rule mining in ontological knowledge bases with amie. *The VLDB Journal*, 24(6):707–730, 2015.

[7] Wille R. Ganter B. Formal concept analysis:mathematical foundations, 2012. Springer Science & Business Media.

[8] H. Hamza, Y. Belaıd, and A. Belaıd. A case-based reasoning approach for invoice structure extraction. 2007.

[9] Vinh Thinh Ho, Daria Stepanova, Mohamed Hassan Gad-Elrab, Evgeny Kharlamov, and Gerhard Weikum. Learning rules from incomplete kgs using embeddings. In *The 17th International Semantic Web Conference*. ceur. ws. org, 2018.

[10] Dmitry I. Ignatov. Introduction to formal concept analysis and its applications in information retrieval and related fields, 2015. Russian Summer School in Information Retrieval.

[11] Michael Stephens James Turk. Jellyfish, a python library for approximate and phonetic matching of strings, `http://github.com/jamesturk/jellyfish`, year = 2018.

[12] Shubham Paliwal, Vishwanath D, Rohit Rahul, Monika Sharma, and Lovekesh Vig. Tablenet: Deep learning based end-to-end model for table detection and tabular data extraction from scanned document images. 2019.

[13] Claudio Antonio Peanho, Henrique Stagni, and Flavio Soares Correa da Silva. Semantic information extraction from images of complex documents. *Applied Intelligence*, 37(4):543–557, 2012.

[14] Najoua Rahal, Maroua Tounsi, Mohamed Benjlaiel, and Adel Alimi. Information extraction from arabic and latin scanned invoices. pages 145–150, 03 2018.

[15] Rohit Rahul, Gunjan Sehgal, Arindam Chowdhury, Monika Sharma, Lovekesh Vig, Gautam Shroff, Ashwin Srinivasan, et al. Deep reader: Information extraction from document images via relation extraction and natural language. *arXiv preprint arXiv:1812.04377*, 2018.

[16] Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. Deepdesrt: Deep learning for detection and structure recognition of tables in document images. In *ICDAR*, 2017.

[17] Frederick Schulz, Markus Ebbecke, Michael Gillmann, Benjamin Adrian, Stefan Agne, and Andreas Dengel. Seizing the treasure: Transferring knowledge in invoice analysis. pages 848–852, 07 2009.

[18] Faisal Shafait and Thomas M Breuel. A simple and effective approach for border noise removal from document images. In *2009 IEEE 13th International Multitopic Conference*, pages 1–5. IEEE, 2009.

[19] Monika Sharma, Shikha Gupta, Arindam Chowdhury, and Lovekesh Vig. Chartnet: Visual reasoning over statistical charts using mac-networks. 2018.

[20] Monika Sharma, Abhishek Verma, and Lovekesh Vig. Learning to clean: A gan perspective. In Gustavo Carneiro and Shaodi You, editors, *Computer Vision – ACCV 2018 Workshops*, pages 174–185, Cham, 2019. Springer International Publishing.

[21] Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In *European conference on computer vision*, pages 56–72. Springer, 2016.

[22] Hai Dang Tran, Daria Stepanova, Mohamed H. Gad-Elrab, Francesca A. Lisi, and Gerhard Weikum. Towards nonmonotonic relational learning from knowledge graphs. In James Cussens and Alessandra Russo, editors, *Inductive Logic Programming - 26th International Conference, ILP 2016, London, UK, September 4-6, 2016, Revised Selected Papers*, volume 10326 of *Lecture Notes in Computer Science*, pages 94–107. Springer, 2016.

[23] L. Yujian and L. Bo. A normalized levenshtein distance metric, 2007. IEEE transactions on pattern analysis and machine intelligence, vol. 29, no. 6, pp. 1091–1095.