Document Classification and Information Extraction framework for Insurance Applications

Ananth Raj GV
American Family Insurance
Madison, USA
agauribi@amfam.com

Dan Dickinson
American Family Insurance
Madison, USA
ddickin@amfam.com

Glenn Fung
American Family Insurance
Madison, USA
gfung@amfam.com

***Abstract*—Document Intelligence is an essential subclass in the field of machine learning. It plays a vital role in insurance applications and other sectors. In this work, we showcase a business application that uses two different but Complimentary techniques: document classification and entity extraction. We also provide an overview of an end-to-end production level system that incorporates deep learning models deployed at scale. The system's backbone relies on trained models carefully analyzed and designed to generalize well on existing and future use-cases. Through empirical evidence, we provide insights into several models trained on our insurance-related datasets and highlight models that have shown good performance across multiple datasets in our real-world insurance setting.**

I. Introduction

Every year, insurance companies consume millions of document images of hundreds of categories, including medical documents, checks, attorney correspondence letters, police reports, and subrogation documents. These documents contain critical information such as personal identifiers, claim numbers, policy identifiers, medical providers, and insured property information that communicate across business workflows and systems. Therefore, these documents primarily drive the insurance claim processing workflows and play essential roles in other business divisions. Being able to classify and extract relevant information from these documents automatically, can significantly improve the efficiencies of numerous business workflows, reduce manual operation costs, enhance the quality and the re-usability of crucial information. We designed, implemented and deployed an automated document classification and information extraction pipeline (see Figure 1, 8, 9) that can democratize this functionality across the enterprise and could be used at scale for many similar use-cases.

Document image classification and information extraction machine learning approaches have been very well researched [1], [2], [5], [7], [8], [10]. However, most of the algorithms are developed based on only a few open-source data sets [3] with limited types of general business documents. In our insurance context, document formats can range from entirely unstructured, semi-structured to fully structured, such as tables and forms, logos, letterheads, and handwritten scripts. Therefore in this paper, we sort general document classification and information extraction methods that have the potential to be generalized to current and future documents. Our proposed methods have three contributions, which we describe below.

The first contribution is the application of our document classification and information extraction pipeline to automate two relevant insurance workflow applications: Medical Bills and Salvage Claims. In the medical bill application, we use deep learning models to first classify the type of medical bill and then extract claimant information. These models will be used to improve the medical bill workflow that handles 300,000 and growing medical bill documents per year during our claims process. In the automobile salvage claims application, we use document classification to identify bank-issued letters of guarantee for loan payment and extract car information. This automation will reduce manual operation costs and improve the processing efficiency of 100,000 total loss claims annually.

The second contribution is the creation of a simple yet effective LayoutLM model [13] for information extraction. Previous work on information extraction from document images are primarily, extensions from Named Entity Recognition (NER) and Conditional Random Fields (CRF). The performance of these techniques is insufficient on the insurance data set, as the documents can be tabular or the sequential pattern is nonexistent. These observations motivated us to use the LayoutLM model that successfully learns and predict the entities and locations within documents, therefore achieving significantly better results than our BiLSTM-CRF [10] and Character Based Sequence Classification baselines. For the document classification task, we altered the ensemble VGG model in [1] to include word vector representations of the text in the document, further improving classification performance.

The third and the last contribution is the deployment of a machine learning pipeline to annotate, train/retrain and deploy these models using a scalable architecture consisting of Airflow Scheduler, AWS API Gateway, AWS Sagemaker, and Elastic Container Service. The last stage of the pipeline involves serving the models as a fast API-based rest application to consume PDF files from an upstream system. Up-steam systems currently post rest requests with PDF files as byte streams. The requests are subsequently processed in 3 stages, Figure 1 - **Document Processing**, **Document Classification** and **Information Extraction**. Document processing involves processing the document into page-level images. The second stage classifies the page images into class labels, which then moves to the last stage - information extraction. This stage

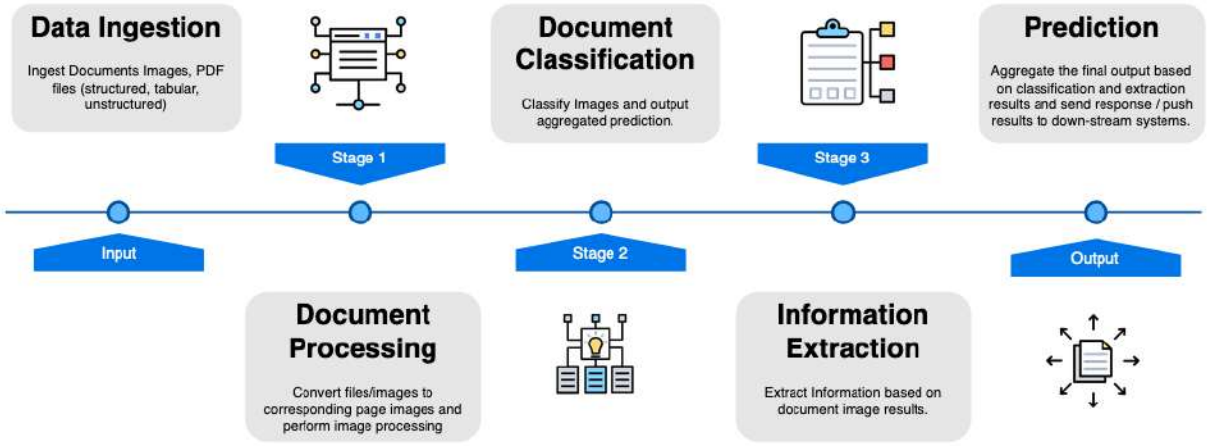

Fig. 1: Document Image Classification and Information Extraction Pipeline

extracts relevant information based on the application. The final output is the prediction of both the classification and extraction model based on the business rules.

II. Related work

Prominent deep-learning-based methods for PDF document image classification tend to be either image-based or text-based. The method described in [1] uses an ensemble of VGG16 models that have been pre-trained on ImageNet to classify documents from the RVL-CDIP data set [3]. The ensemble consists of five separate pre-trained VGG16 models, each trained on the RVL-CDIP data set using either left, right, top, bottom, or no crop of the image, and achieves an accuracy of 92.2%.

Information extraction from documents has traditionally been treated as a text-based task, using model architectures such as BILSTM-LSTM and BILSTM-CRF [7]. Techniques that further incorporate structural information of the documents include graph convolutional networks [8], which encodes the document structure as a graph with regions of text as nodes and edges encoding information such as aspect ratios and distance between the regions being connected.

Chargrid [5] and BERTgrid [6] are two techniques that encode the spatial structure of the documents as 2D grids of characters and contextualized token vectors respectively. Chargrid uses a one-hot encoded vector to represent the characters, whereas BERTgrid represents partial word tokens using contextualized vectors from the popular BERT model [9]. Both models then employ a convolution encoder-decoder architecture that predicts bounding box coordinates as well as the character or word token within the box.

LayoutLM [13] is a pre-training method that combines the text of the document with the layout of the 2D document, also employing BERT representations of word tokens, as well as an image embedding for bounding boxes around identified words. These learned representations can be used for downstream information extraction and document classification tasks and have been shown to outperform many previous SOTA benchmarks in this domain.

III. Applications

A. The Medical Bills application

Our insurance company has been ingesting medical bills from external health institutions for automobile insurance casualty claims review and re-pricing. Streaming in from snail mails and emails throughout the day, medical bills consist of three types: HCFA (Health Care Finance Administration), UB (Uniform Billing), and Non-standard bills. Figure 2(a)-(c) shows a sample image of each type of medical bills. All medical bills are mostly in a tabular format, with HCFA and UB having a type-specific layout. Other medical bills table structures can be arbitrary. Currently, a team of operational support staff manually classifies medical bills into three classes and extracts relevant content such as claim identifiers, date of birth (DOB), names, addresses, patient identifiers using OCR or other template matching algorithms. Another team of operational support staff is responsible for auditing the classification and extraction results. The classified documents and extracted information of the medical bill will then be stored and reused. Approximately 300,000 medical bills need to be processed per year, and downstream claim processing workflows will reuse the classified and extracted information. This paper intends to propose generic machine learning approaches to automate both medical bill classification and information extraction. Our proposed method can significantly reduce labor cost, task turn-around time, and overcome the limitations of brittle template-matching algorithms that will fail in the case of new document formats.

B. The Salvage Claims application

Personal Automobile Insurance line processes 80,000 to 100,000 total loss claims per year. A total loss automobile claim, or a salvage claim, is filed when the damaged automobile's repair cost or salvage cost exceeds the insured

(a) Medical HCFA

(b) Medical UB04

(c) Medical Other

(d) Salvage LOG

(e) Salvage Sales

(f) Salvage Other

Fig. 2: Samples of (a) HCFA (Health Care Finance Administration) Medical Bill (b) UB (Uniform Billing) Medical Bill (c) Other Medical Bill, (d) Salvage Claim Letter of Guarantee (e) Salvage Claim Sales Receipt (f) Other Salvage Claim document

value. One typical workflow to process a salvage claim is to collect a letter of guarantee (LOG) issued from banks that previously issued a lease or loan for the damaged automobile. LOG guarantees that the insurer's reimbursement will pay the bank first and then the automobile titleholder. Information such as claim id, vehicle identification number(VIN), year, make, and car model will be extracted from the LOG to complete the salvage claim processing. We typically receive 3500 to 4000 salvage claims emails per month, with 40 documents per hour at peak time. The email attachments consist of LOG, Sales Receipt, and other types of documents (Figure 2(d)-(e)). The LOG content is usually semi-structured with a block containing claim id, vin number, year, make, and car model. Sales receipt and other documentation associated with salvage claims can be both tabular or unstructured. One to two full-time employees are responsible for classifying documents attached to the email and extract relevant information. The manual process is quite repetitive and expensive, and the salvage claims usually involve low monetary values and need little investigation. Therefore we also intend to use our proposed approach to eliminate human labor in this particular use case to achieve the long-term goal of completely automating thousands of claims a year.

IV. Methods

A. Document Image Classification

We describe three models and how we alter them to achieve high accuracy for both the applications we are trying to solve. We start with the baseline VGG model [11] and compare the accuracy with modified architectures Figure 3. The baseline VGG model performs well on classification tasks, but recent new ensemble models that we use have proved to provide better accuracy scores than the baseline model.

1) VGG Baseline: VGG16 [11] is a popular class of Deep Learning Convolutional Neural Networks that have been widely used for Image Classification. VGG16 consists of 16 layers of a combination of convolutional, max pooling, fully connected layers, and a 1000-way softmax classifier. This model achieves an accuracy of 92.7% on the ImageNet dataset. We use this model pre-trained on ImageNet and train it on our custom dataset using the concept of transfer learning. We change the last classification layer of the model to output the softmax scores over three classes(LOG, SALES, OTHER) for Salvage and (HCFA, UB 04, OTHER) for Medical application.

2) Ensemble VGG: VGG models can be utilized in other ways to classify images. Recent methodologies have introduced the concept of intradomain transfer learning. The idea is to train multiple VGG models for specific regions and use them as feature extractors for an ensemble model [1]. We train five such region-specific models targeting specific regions - top, bottom, left, right, and holistic, respectively. We first train a base VGG model on the RVL dataset [3] consisting of 400,000 grayscale images in 16 classes. The trained model is then used as the starting point for transfer learning to extract region-specific features for our application. We resize the images from our dataset to (256, 512) dimensions for top and bottom regions and (512, 256) dimensions for the left and right regions. After fine-tuning these models, we derive five different models. They are used as feature extractors to our final ensemble model, which is a couple of fully connected layers and a softmax layer. Input to the ensemble model is horizontally stacked features from all the region models with a 3-way softmax classifier as the output.

3) Ensemble VGG + Text Features: The final model we trained and currently use in production incorporates text-level features along with image features. This model is sensible because many insurance domain documents are distinctly similar at a template level. Hence, it is sensible to infer that incorporating text features when training a model would boost performance as the model attends to an image's visual characteristics coupled with text-level information. We use the same architecture as the Ensemble VGG classifier. However, a small change incorporates extracting a 300-dimensional document text embedding using Fasttext [15] and appending it to the 1000 dimensional VGG features before passing it through Linear layers and a Softmax layer Figure 3. The output of the trained region models is stacked together as input to the following ensemble model. We observe better accuracy scores Table I and use this model in our production environment.

B. Information Extraction

Information Extraction for document images is a widely sought out area in the domain of machine learning. In particular, insurance industries are researching and developing many models that can be generalized easily across multiple

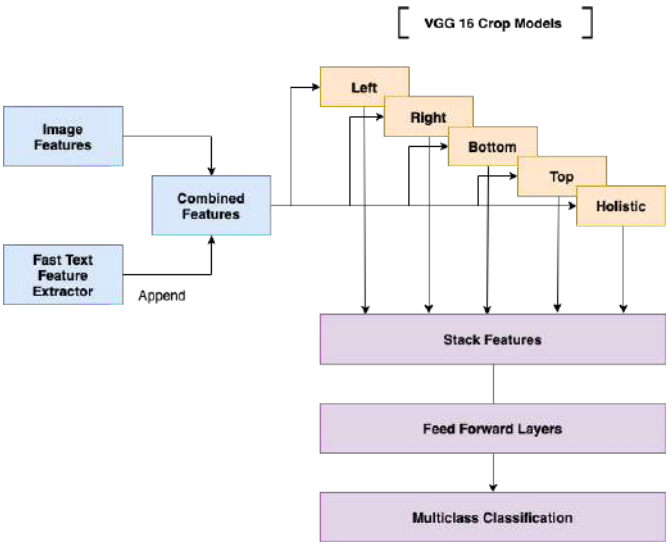

Fig. 3: Ensemble VGG + Text Features Classification Model

use cases. The main problem of information extraction is generalizing the model to have consistent performance across various templates(articles, tabular, diagrams, and numerical data). In general, documents have various templates, which makes developing an end-to-end model to extract information a difficult task. We use a token level and character based BiLSTM-CRF as our baseline and discuss some limitations of this method. Finally, we show how a LayoutLM model [13] addresses those issues by generalizing well irrespective of the dataset and show improved accuracy scores in our application.

1) BiLSTM-CRF Baseline: With the advent of LSTM networks in deep learning, NER problems have been tackled using LSTMs, BiLSTMs, and BiLSTM-CRFs (Figure 4). The CRF layer has proven to be a useful tool in the NER task domain, due to its ability to model the semantic structure of the ordering of tokens present in the text. However, when modeling documents like semi-standardized forms, there is not much semantic structure present in the word tokens (as opposed to say a wordy legal contract, as in [7]). A noticeable limitation of the CRF model is that it is designed to detect token transitions in a long text sequence. It is not very effective in distinguishing the representations of the token itself.

2) Character Based Sequence Classification: To overcome the limitations of the CRF model, we use a character-based BiLSTM sequence model. BiLSTM networks can be extended to a character-level [12] where the input to the model is character sequence and not individual tokens. In our use case, examples of the Claim number and VIN could be 1234-56789-123 and JPAG764589 respectively. Framing this as a character level sequence to sequence model, the model tends to learn the character's internal representation in tokens. Input to the model are character encodings, and the output is processed to capture the longest subsequence of class labels for all predicted fields. Figure 5 shows a hypothetical output of the model: 0001111002222000333. Here, 1 would represent the claim id subsequence, 2 would represent VIN, and so forth. In this

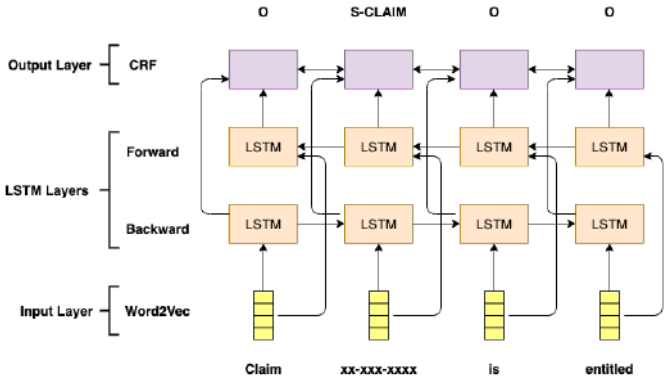

Fig. 4: NER BiLSTM-CRF Baseline Model

workflow, OCR is used to extract text from the document. This process is not perfect, and can result in incorrectly extracted text (e.g. a "1" instead of an "I"). One benefit of a BiLSTM model trained on character sequences is that it generalizes well on the dataset despite these errors in the OCR module. We also experimented with an additional CRF layer to this model with no significant performance improvement.

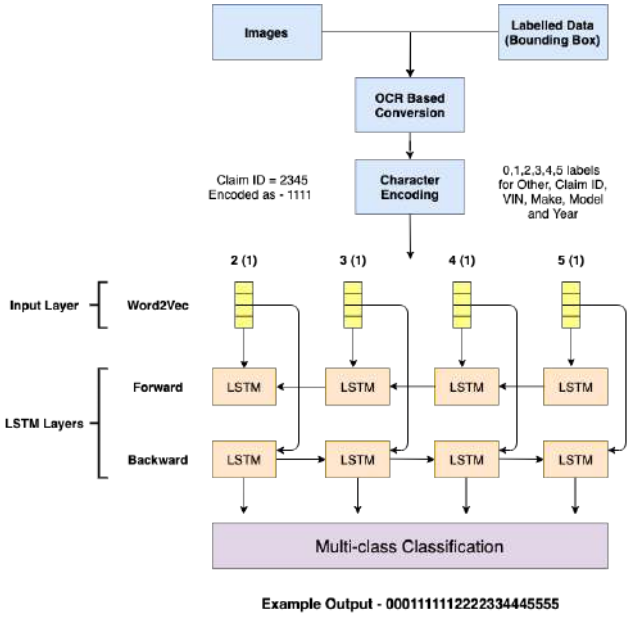

Fig. 5: Character Based Sequence Classification Model

3) LayoutLM Model: Text-based information extraction models do not account for layout information which becomes a major setback for semi-structured documents. To accommodate the visual characteristics and textual information, we implemented the LayoutLM model [13] for token classification. LayoutLM model being a BERT variant is capable of harnessing the power of BERT for learning contextual representation of text appearing in documents. Additionally, the model also learns the location of entities from the coordinate bounding boxes passed as input to the model. This substantiates the high accuracy scores we see in our experiments. We

use the pre-trained LayoutLM-Base, Uncased model (v1) for all our experiments. Document images are processed using pytesseract to extract text and bounding box coordinates, as input to the LayoutLM model. We fine-tune the model under the sequence labeling task setting for 10 epochs and validate it with our held-out test set.

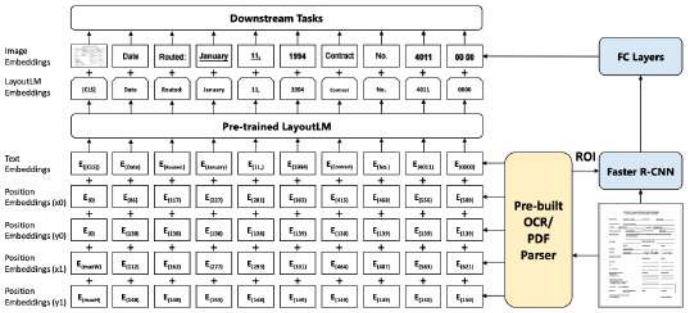

Fig. 6: LayoutLM Model [13]

V. Experiments and results

Data	Model	Auc(ovr)	Auc(ovo)	Acc
Salvage	VGG Baseline	0.977	0.976	94.72
	ENS VGG	0.998	0.998	97.47
	ENS VGG + TF	**0.999**	**0.999**	**99.26**
Medical	VGG Baseline	0.959	0.964	82.41
	ENS VGG	0.978	0.979	86.66
	ENS VGG + TF	**0.999**	**0.999**	**98.13**

TABLE I: Accuracy and AUC Scores for the classification models, VGG, ENS VGG (Ensemble VGG), and ENS VGG + TF (Ensemble VGG with Text Features). The abbreviations ovr and ovo denote "one-vs-rest", and "one-vs-one", respectively.

A. Labels And Annotations

We adopt a Human in the Loop learning approach for both document image classification and information extraction. We have built an annotation tool as part of our machine learning pipeline to label document classes and to annotate bounding boxes around claim id, name, addresses, DOB, and other information(Figure 9). The tool is developed to have built-in pytesseract OCR support to enable easy click and annotate features. The tool thus captures the labels + bounding box coordinates for each entity that is annotated.

B. Datasets and Experiment Setups

For the Medical Bills application, we collect 1500 Medical bills consisting of 500 examples from each of the 3 classes - HCFA, UB04, and Non-standard bills, along with their class labels. We labeled and annotated 700 medical documents for extracting the relevant fields. For the Salvage Claims application, we collect a dataset that consists of 2,000 documents each of Letter of Guarantee, Sales Tax, and Other class. We labeled and annotated 700 salvage documents for extracting the relevant fields. Both the applications first classify these documents and then extract information: claim number, name, DOB, and patient id for medical bills; claim number, VIN, make, model, and year for salvage claims. The annotated bounding box coordinates are processed as input to the extraction model. We train the classification model and extraction model on p3.2x large GPU instances on AWS.

C. Document Pre-processing

Documents are generally in PDF format and are converted to respective page-level images using the python package pdf2image. We enhance the images to 300 dpi, convert them to grayscale, and retain the images' original dimensions. Images are then processed using PyTesseract OCR to extract text from images. To train our classification and extraction model, labeled images are divided into training, validation, and test data. We follow an 80%, 10%, 10% split in our data sets. Held out test sets are unique documents not seen in the train set. We incrementally report our results, starting with base models towards models with better performance.

D. Document Classification result

The evaluation metrics used for the classification model are accuracy and AUC score. Since this is a classic multi-class classification problem, we compute the AUC of each class against the rest, also called OVR, and compute the average AUC of all possible pairwise combinations of classes, also called OVO. From Table I, we can see that the ensemble VGG model trained on region features and text features (combined) outperforms the models trained otherwise and generalizes well across both medical and salvage applications. We depict a 3% increase in accuracy and a substantial improvement in AUC scores in our final model compared to the Salvage dataset's baseline model. There is a considerable improvement in accuracy and AUC scores for medical bills as well. Medical bills being highly structured, we can conclude that text-level features play an important role in model classification. We also report the confusion matrix for our classification model as shown in Figure 7. It is evident that the model makes negligible misclassifications in both the applications.

E. Information Extraction result

The evaluation metrics used for the extraction model are precision, recall, and F1 scores. We report field-level metrics to get a detailed view of how the model performs in extracting information. Reporting these metrics to the business enables them to decide the trade-off in precision, recall, and set a threshold to loop in a human for low confidence results.

1) NER BiLSTM-CRF Baseline: Based on our experiments on our insurance dataset, we observe that, though BiLSTM-CRF is the go-to option for Information Extraction, these models may not always perform well on all datasets. Our

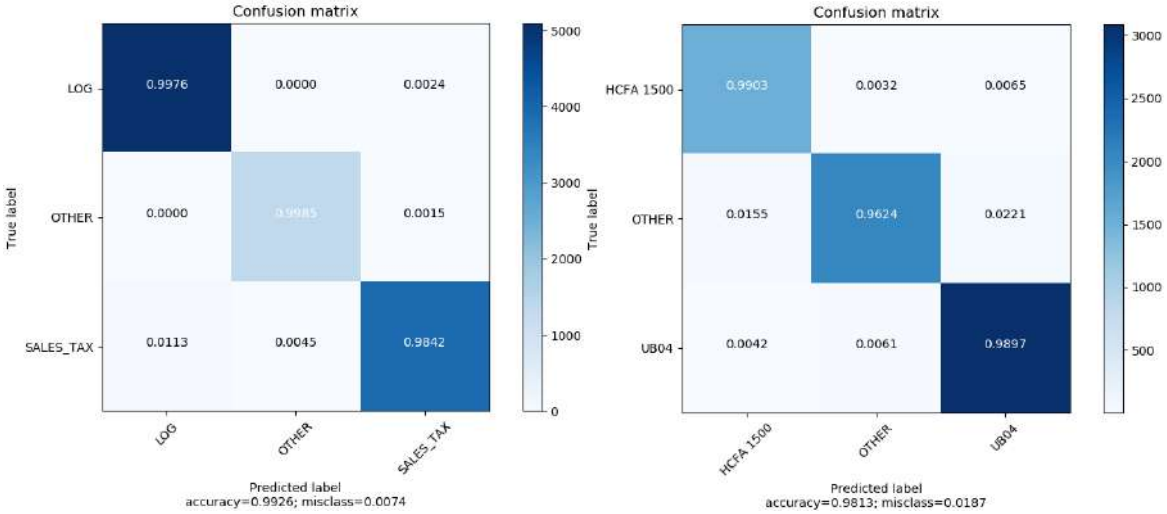

Fig. 7: Confusion Matrices of ENS VGG + TF for Salvage Claims (left) and Medical Bills (right) applications.

Application	Field	Precision			Recall			F1		
		CRF	Char	LayoutLM	CRF	Char	LayoutLM	CRF	Char	LayoutLM
Salvage	OTHER	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	CLAIM ID	0.30	0.89	**0.92**	0.38	**0.96**	0.94	0.34	0.92	**0.93**
	VIN	0.45	0.72	**0.86**	0.71	0.65	**0.81**	0.55	0.68	**0.83**
	MAKE	0.17	0.57	**0.85**	0.20	0.64	**0.70**	0.18	0.60	**0.77**
	MODEL	0.35	0.67	**0.87**	0.50	0.61	**0.85**	0.41	0.64	**0.86**
	YEAR	0.50	0.51	**0.90**	0.50	0.57	**0.92**	0.50	0.54	**0.91**
Medical	OTHER	1.00	0.96	1.00	1.00	0.97	1.00	1.00	0.96	1.00
	NAME	0.18	0.53	**0.95**	0.22	0.90	**0.93**	0.20	0.67	**0.94**
	DOB	0.41	0.65	**0.93**	0.33	**0.89**	0.86	0.37	0.75	**0.89**
	CLAIM ID	0.33	0.50	**0.88**	0.29	0.93	**0.93**	0.31	0.65	**0.90**
	PID	0.40	0.40	**0.92**	0.36	**0.91**	0.89	0.38	0.55	**0.90**

TABLE II: Information Extraction Precision, Recall, and F1 scores for CRF (BiLSTM-CRF Baseline), Char (Character Based BiLSTM) and LayoutLM

dataset is a combination of articles, tabular, and data of other forms where the tokens to be classified are not a sequence of tokens but relatively sparse words in a long text sequence. Textual information is derived from OCR processing, so there is always a setback in our pipeline that PDF documents have probable low-quality scans. Hence text derived from OCR processing is often partial, and tokens to be categorized could have different rendering patterns based on tabular or article documents. We observe low precision and recall scores for our dataset under this methodology, Table II.

2) Character Level Bidirectional LSTM Sequence Classification: This model tends to learn a better representation among fields - name, DOB, claim number, PID in the medical application, and claim id, year, model, make, and VIN in the salvage application. Unlike the BiLSTM-CRF model, we can conclude that the Character-based sequence classification model generalizes well on both medical and salvage applications. The model performs well in identifying and extracting fields because it learns the positional occurrence of specific fields like claim number with respect to others, such as make, model, or VIN during training. It also understands that specific fields are only numeric, only characters, or alpha-numeric. Since the model captures this information, the recall for claim id, DOB and VIN is high compared to LayoutLM as these fields are often few characters in length and always have a generic pattern. This helps the model generalize well irrespective of the structure of the dataset. We observe good precision and recall scores for our dataset under this methodology, Table II.

3) LayoutLM: We can clearly see a boost in performance with the LayoutLM model. This boost can be attributed to the fact that the model jointly learns the contextual information of text along with its location in the document. This is highly beneficial for structured(medical bills) and semi-structured(salvage) document types. The model not only learns the internal representation of tokens but also learns the styling associated with text such as fonts and shapes. This adds to the model accuracy often because claim number is generally bold and numeric in scanned documents. Text-based models do not have the power to capture these visual characteristics that play an important role in document processing. We can see that the precision and recall for most fields that we want to

capture is well over the 85% margin, thus making the model very accurate to capture relevant information.

VI. Deployment

A. Phase 1 - Container-as-a-Service deployment

We initially deployed and served classification and information extraction models in Amazon Web Service for both salvage and medical bills as standalone applications. We designed the production serving architecture by considering the incoming traffic frequency, volume, latency, and computing resources (e.g., CPU, GPU, RAM) for processing documents.

Both the medical bills and salvage applications have the requirement of near real-time processing. Documents are streamed during the day with peak traffic of 40 documents per hour in the salvage application and an estimated number of daily documents of around 1000 in the medical bill's application. It usually takes 4 to 8 seconds to process a one-page PDF document. However, it can take up to 15 to 20 seconds when the incoming PDF document contains multiple pages. We opted to run our document image classification and information extraction as an Elastic Container Service (ECS) Fast API server running on AWS Fargate. The ECS tasks are configured with a computing capacity of 30GB RAM and 4 vCPU's to handle the requirements for processing each document. We initially trained the models using GPU and deployed them separately on the CPU, making this a manual effort for data scientists.

B. Phase 2 - Machine Learning Pipeline

After Salvage and Medical document type applications were deployed to production, we realized the need to set up a machine learning pipeline to incorporate additional business use cases and doc-types. Within our firm, we have more than 30 doc-types and it is time-consuming to setup up the individual training and deployment strategies. In order to iterate fast in deployment, we developed a machine learning pipeline with Human in the Loop. As part of the pipeline, we enhanced the annotation tool as our front-facing Indexing UI. Any business can use the UI to train/retrain custom models. The Indexing UI allows uploading or streaming of documents to or from s3 bucket. After the user annotates few documents, he can trigger the training job with a single click. The backend comprises a pipeline orchestration designed using Apache Airflow. Once the training task is triggered as an API call to the backend, airflow runs a couple of stages sequentially such as pulling the data from s3 for which the user triggered training, pre-processing the documents, running OCR, training a custom model, and logging the metrics to MLflow. After training is complete the model is tracked by MLflow model registry and information such as model ID and metrics are sent back to the UI. The user has the freedom to now finetune the model by annotating additional documents or deploying a Fast API-based ECS rest end-point to serve the model. Once the model is deployed, the business has the flexibility to integrate this AI solution with their downstream workflows. This deployed strategy is an attempt to provide a self-service-based AI solution to any business use case without the active involvement of data scientists.

A particular business application such as salvage uses our AI-powered solution as follows - Incoming mail with pdf attachments are forwarded to the rest endpoint serving the model they created as part of the initial training workflow. The model predictions are sent back to the business. They use the model predictions to validate against the claim's center datalake. If the model predicted field such as claim number matches the claim's center, additional predictions are updated as metadata against the claimant info. If there was no match, the model prediction on the claim number was either wrong or there was some error in the OCR. Such documents are pushed to the UI for a human to audit. The business has dedicated auditors to verify this information and correct the model predictions on the UI, which then become the feedback for future retraining. Retraining is scheduled to run on a timely basis as part of our backend pipeline.

Currently, the salvage application which is one among the business use case we are addressing has reached an accuracy of 95% on both classification and extraction tasks with Human in the Loop. This alleviates the human effort to manually classify and extract information visually for each document.

VII. Conclusion and future work

Automatic document image classification and information extraction (DOCIE) are two frequently occurring tasks in enterprises such as insurance companies. Millions of documents need to be continuously organized and indexed on an on-going basis. In this paper, we proposed an end-to-end system that accomplishes this chain of tasks by leveraging both state-of-the-art computer vision and language models. We have introduced two innovative and impactful applications for document image classification and information extraction: medical bill classification and claimant information extraction for casualty lines, and the other is classifying a type of contract issued by banks and car information extraction for automobile salvage claims. As seen from our results, the extraction model we use - LayoutLM, is seen to generalize well irrespective of the structure of the incoming documents. The model may hence do a good job addressing any business use case in the foreseeable future. The design of our machine learning pipeline accounts for horizontal and vertical scalability.

We are also exploring some of the newly proposed zero shot [14] and few shot [15] information extraction methods. As many more types of document images from various business functions will be automated using our machine learning pipeline, we will attempt to incorporate zero shot and few shot learning capabilities in future. We plan to generalize our models to extract comprehensive types of personal identification information and entities of interest from a wide range of insurance documents, including personal checks, purchase receipts, tax documents, medical records, subrogation letter, etc. Building large pre-trained data sets using larger sets of insurance-specific data sets can also be beneficial. We can

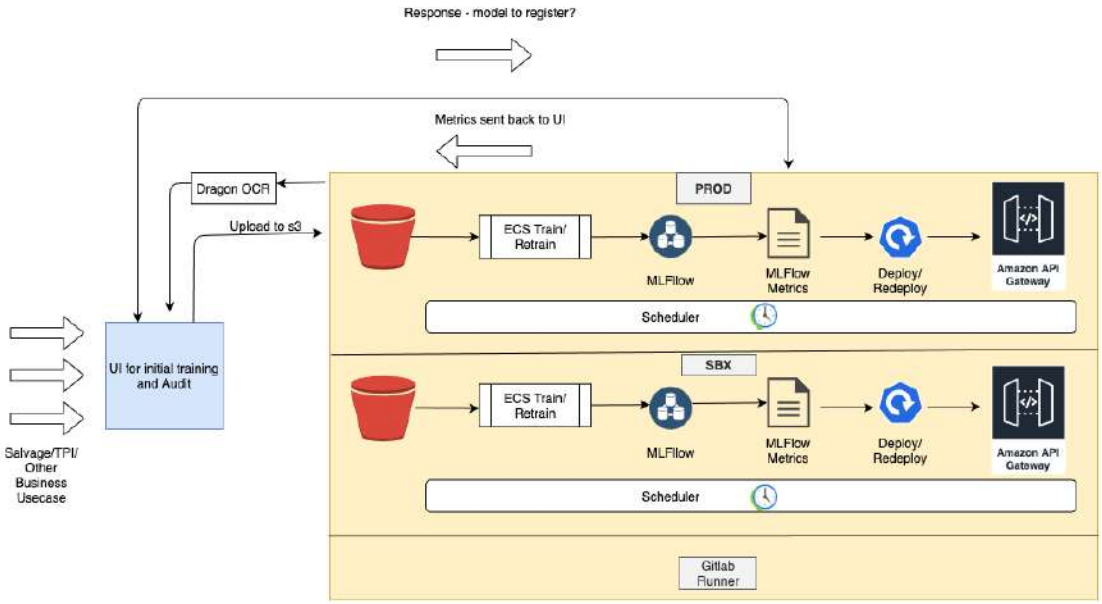

Fig. 8: Backend ML Pipeline - Airflow Scheduler, Multiple Stages to consume documents, pre-process, train/retrain, deploy the model with scaling

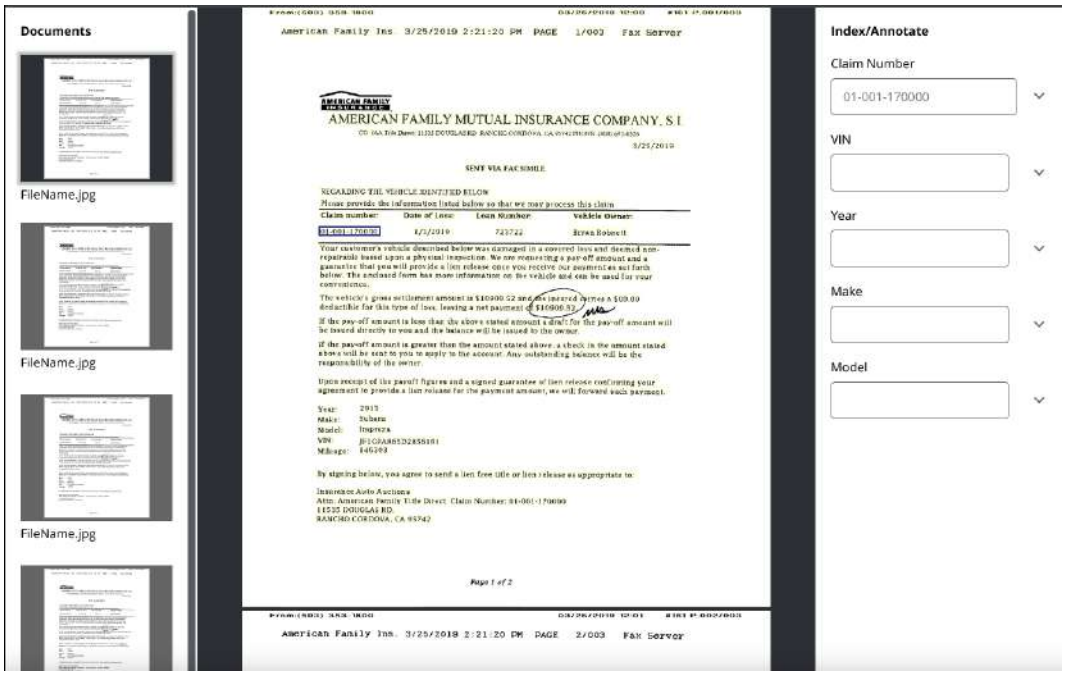

Fig. 9: UI for business to annotate, audit documents, trigger train/retrain, Human in the Loop and Model Registry

use transfer learning, few shot learning, or active learning to reduce labeling efforts for specific applications.

REFERENCES

[1] Das, Arindam and Roy, Saikat and Bhattacharya, Ujjwal and Parui, Swapan K, "Document Image Classification with Intra-Domain Transfer Learning and Stacked Generalization of Deep Convolutional Neural Networks", 24th International Conference on Pattern Recognition (ICPR), IEEE, August 2018.

[2] Le Kang and Jayant Kumar and Peng Ye and Yi Li and David S. Doermann, "Convolutional Neural Networks for Document Image Classification", 22nd International Conference on Pattern Recognition, ICPR, IEEE, 2014.

[3] Adam W. Harley and Alex Ufkes and Konstantinos G. Derpanis, "Evaluation of deep convolutional nets for document image classification and retrieval", 13th International Conference on Document Analysis and

Recognition, ICDAR 2015, Nancy, France, August 23-26, 2015.
[4] Adam W Harley and Alex Ufkes and Konstantinos G Derpanis, "Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval", International Conference on Document Analysis and Recognition, ICDAR, 2015.
[5] Katti, Anoop R and Reisswig, Christian and Guder, Cordula and Brarda, Sebastian and Bickel, Steffen and Höhne, Johannes and Faddoul, Jean Baptiste, "Chargrid: Towards Understanding 2D Documents", Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2018.
[6] Timo I. Denk and Christian Reisswig, "BERTgrid: Contextualized Embedding for 2D Document Representation and Understanding", arXiv, 2019.
[7] Ilias Chalkidis and Ion Androutsopoulos, "A Deep Learning Approach to Contract Element Extraction", JURIX, 2014.
[8] Xiaojing Liu and F. Gao and Q. Zhang and Huasha Zhao, "Graph Convolution for Multimodal Information Extraction from Visually Rich Documents", NAACL-HLT, 2019.
[9] Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina, "BERT", Proceedings of the 2019 Conference of the North, Association for Computational Linguistics, 2019.
[10] Zhiheng Huang and Wei Xu and Kai Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging", arXiv, 2015.
[11] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", arXiv, 2014.
[12] Mourad Gridach, "Character-level neural network for biomedical named entity recognition", Journal of Biomedical Informatics, 2017.
[13] Xu, Yiheng and Li, Minghao and Cui, Lei and Huang, Shaohan and Wei, Furu and Zhou, Ming, "LayoutLM: Pre-training of Text and Layout for Document Image Understanding", Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2020.
[14] R. Cao and P. Luo, "Extracting Zero-shot Structured Information from Form-like Documents: Pretraining with Keys and Triggers", AAAI, vol. 35, no. 14, pp. 12612-12620, May 2021.
[15] Yang, Y., and Katiyar, A. (2020). "Simple and Effective Few-Shot Named Entity Recognition with Structured Nearest Neighbor Learning." ArXiv, abs/2010.02405.