DEEP LEARNING DIGITIZATION GRAPH NEURAL NETWORKS OCR INFORMATION EXTRACTION

ID Card Digitization and Information Extraction using Deep Learning - A Review

by **Vihar Kurama** 7 months ago

14 MIN READ

Table of Contents

Introduction

In this article, we will discuss how any organisation can use deep learning to automate ID card information extraction, data entry and reviewing procedures to achieve greater efficiency and cut costs. We will review different deep learning approaches that have been used in the past for this problem, compare the results and look into the latest in the field. We will discuss graph neural networks and how they are being used for digitization.

While we will be looking at the specific use-case of ID cards, anyone dealing with any form of documents, invoices and receipts, etc and is interested in building a technical understanding of how deep learning and OCR can solve the problem will find the information useful.

Deep Learning and Digitization

Many organisations during their onboarding procedures, to have adequate amount of information about their customers, requires customers to submit some documents that they could use to verify their identity and get relevant details about them. A few examples include banks and insurance companies. This process is usually time consuming and prone to errors since it is done manually at a lot of places. The customer is expected to submit a digital copy of the documents which a manual reviewer will review, identify if it is fake, extract information like name, address, etc and enter it into a data entry software.

As deep learning approaches and OCR technologies have progressed, semi or fully automated solutions relating to physical document information extraction are seeing a wider adoption. Here are a few reasons why digitization of information is becoming prevelant -

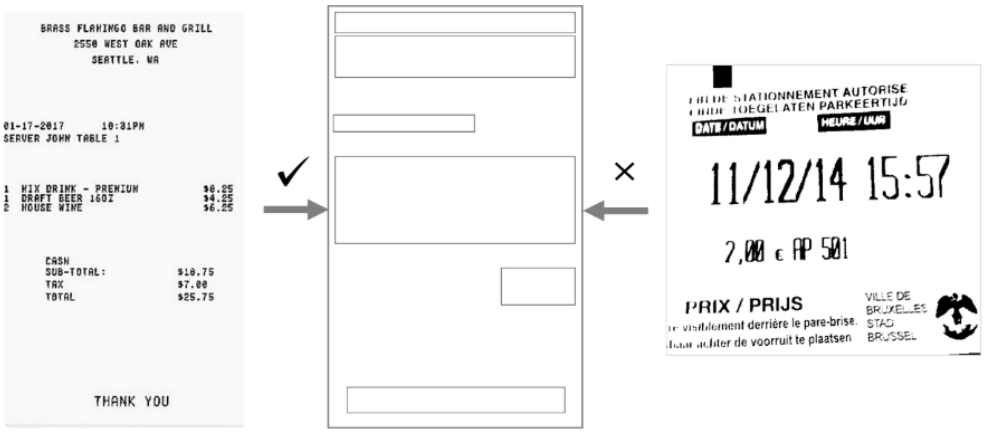
1. ***Information extraction*** - We can capture all the information provided on the ID card and push that data as a unique source for further use. All the information pulled from the captured ID card will be in a simple text/numerical format. This helps to maintain data in an organized

fashion and facilitates any sort of verification or registration process.

2. ***Greater speed and efficiency*** - Digitizing ID cards can save a lot of time and money for businesses and organisations. It takes a couple of seconds to simply scan an ID card and retrieve all the data from it. This shift towards fast digital processes instead of manual entry and review is enabled by deep learning based approaches (discussed later).
3. ***Records data without error*** - With the advancement in technologies and computational power, machines are now capable of capturing the data without many errors. The chances of human error can be reduced by automating repetitive tasks and allowing humans to review document information on the final stages of the information extraction pipeline.
4. ***Integrates easily into any system*** - Digitized solutions can be easily integrated into any system. For example, the model that is trained to identify information from a particular ID card can be deployed on a website where users upload the images in bulk, or it can be used in mobile phones where users click on images and thereby, the information is extracted.

Challenges and shortcomings

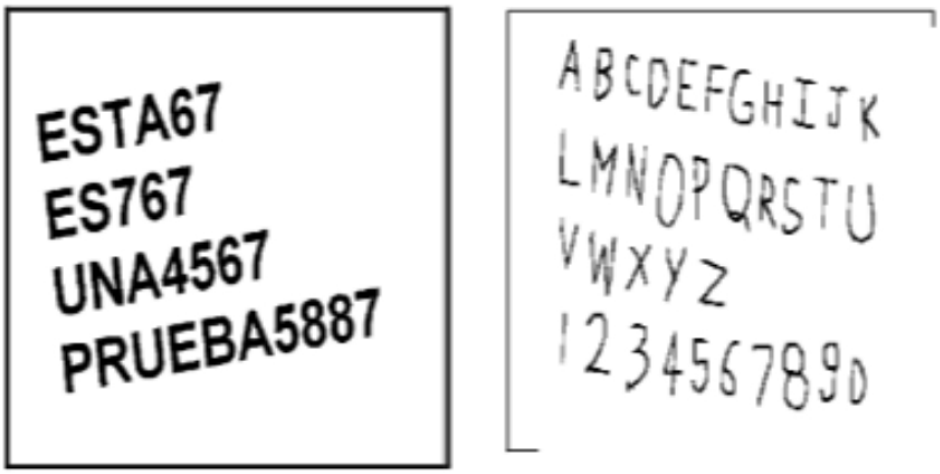
Deep learning has been a remedy to many automation problems, but there are still several challenges researchers and developers face trying to build a perfect model that possesses commendable quality and outputs high accuracy. There can be a good many physical and technical errors, out of which a few are discussed below.



source

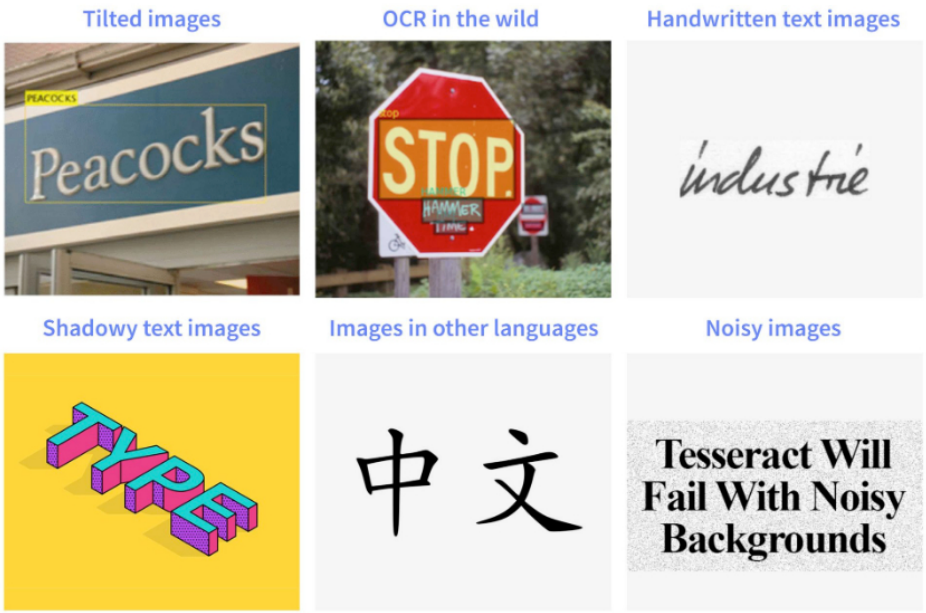
Various Designs, Templates and Multilingual Environments - There are several fonts, designs, and templates for different types of ID cards. Characters of various fonts have large within-class variations and form many pattern sub-spaces, making it difficult to perform accurate recognition when the character class number is large. If at all the model is trained with a random set of images, there is a high probability of the model to underperform. Sometimes few ID cards are printed in different languages. In multilingual situations, capturing information from the scanned documents stays as a primary

research issue since information in complex symbolism is more troublesome.



source

Orientation and Skewness(Rotation) - Scanned documents or ID cards are usually parallel to the plane of the sensor. But when the images are manually captured with the cameras or any digital sorts, they have different issues like Orientation and Skewness. Cell phones have an advantage with orientation sensors. They can recognize whether the device is tilted and when twisting happens, they can forbid clients to take pictures. Skewness is basically the degree of angle in which the ID cards are captured. If the skewness is greater then the model will show some poor results. However, there are several techniques to overcome this problem like Projection Profile, RAST algorithm, Hough transform, methods of Fourier transformation, etc.



Scene Complexity - Usually, scene complexity is defined by the environment

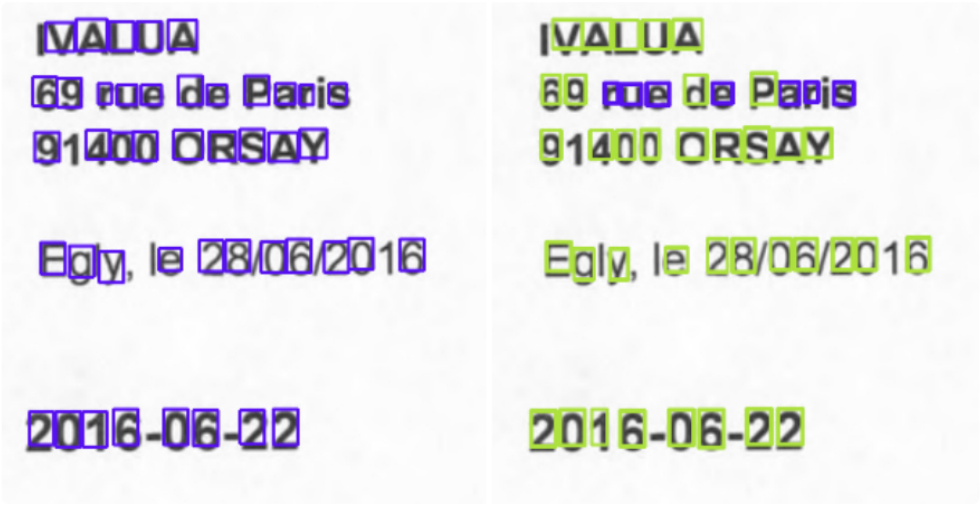
the images are captured in. The complexity depends on several factors, say, uneven lighting, contrasts, tilting(position in which the images are captured), etc. The information captured has comparative structures and appearances to the text which makes processing the image challenging. Hence, to overcome this, we need to make sure the image is preprocessed before training or labeling the ID cards for information extraction. For lighting conditions, we can also use filters and enhancers which highlights the text and makes it easier for the model to process the image.

Digitization and OCR - Traditional Methods

Let's look at how deep learning is used to achieve a state of the art performance in extracting information from the ID cards.

Object Detection

Using popular deep learning architectures like Faster-RCNN, Mask-RCNN, YOLO, SSD, RetinaNet, the task of extracting information from text documents using object detection has become much easier. Here the models are trained on characters which are then recognized as objects in the images. Below is an image that clearly portrays the identification of text from images done using object detection.

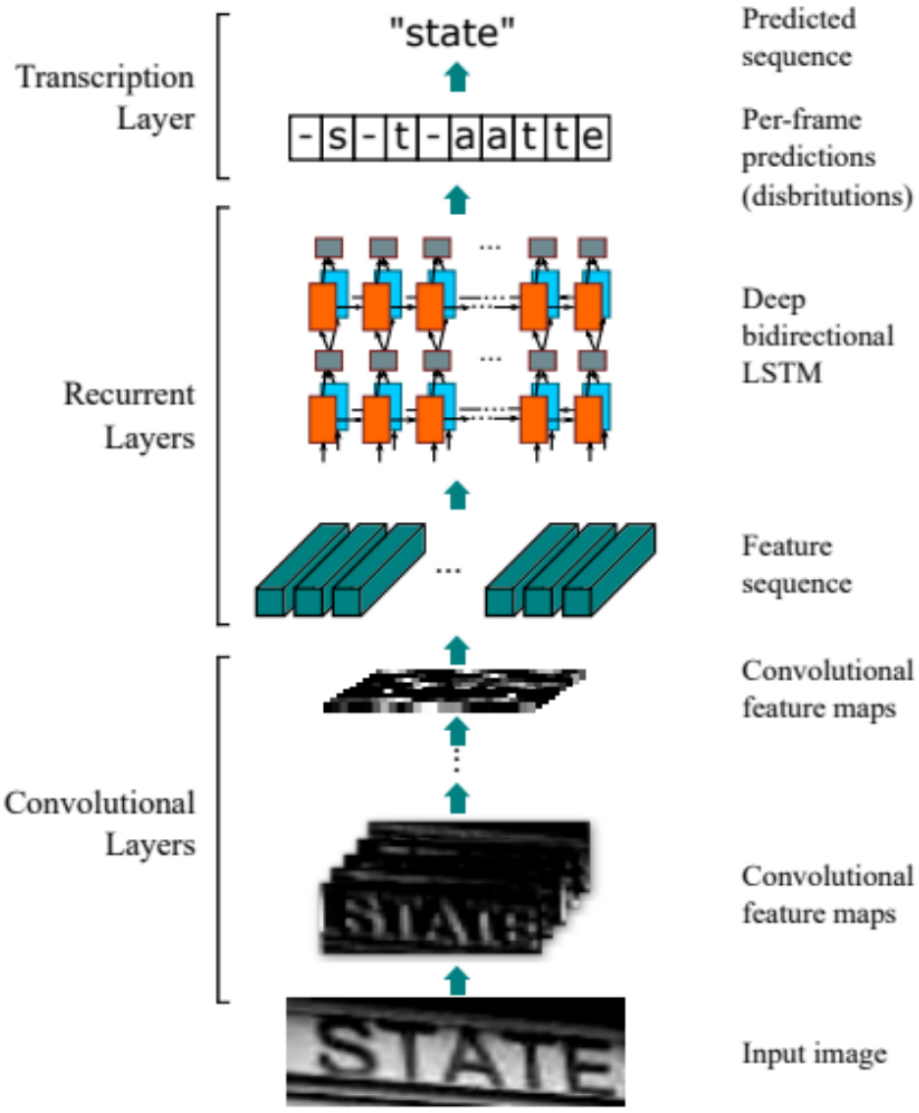


The output of the network consists of individual characters that need to be concatenated as text. Say, if a single character is mispredicted, the error does not affect the entire text.

Convolutional Recurrent Neural Networks

Recurrent neural networks are well known for processing text data. But for extracting information from ID cards, we need an intersection of processing

both images(which are captured) and text(which needs to be identified). To achieve this, the CRNN was introduced in the year 2015. In CRNNs, the first level has a basic fully convolutional network. The next level of the network is defined as a feature layer, and is divided into “feature columns”. The feature columns are fed into a deep-bidirectional LSTM which outputs a sequence and is intended for finding relations between the characters. Below is an image explaining the flow.



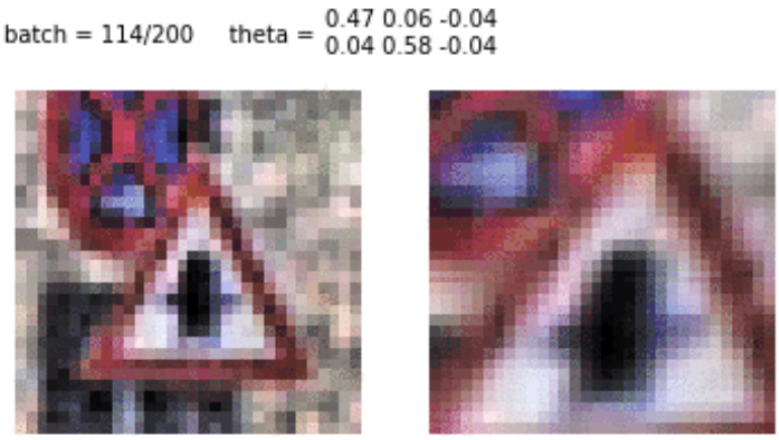
Source: https://arxiv.org/pdf/1609.04243.pdf

Results **-** CRNNs were initially used for music classification and then slowly became a tool for text recognition and classification. This network achieved the accuracy of 77% in the classification task. The CRNNs depth used in the research was up to 20 layers and was trained on 2500 samples.

STN-OCR Network

Spatial transformer network apply affine transformations to removes the spatial variance in the images. It learns what part of an image is the most

important and accordingly scales or rotates the image to focus on that part.



source

STN-OCR is a semi-supervised neural network and consists of a localisation net, a grid generator and a sampler. The localisation net takes an input image and gives us the parameters for the transformation we want to apply on it. The grid generator uses a desired output template, multiplies it with the parameters obtained from the localisation net and brings us the location of the point we want to apply the transformation at to get the desired result. A bilinear sampling kernel is finally used to generate our transformed feature maps. These final transformed feature maps are put through our recognition nextwork to predict the text in our input images.

Below is the network architecture for the STN-OCR network that identifies information from images.

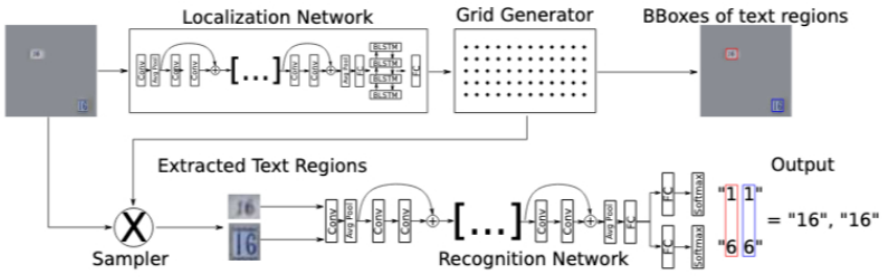


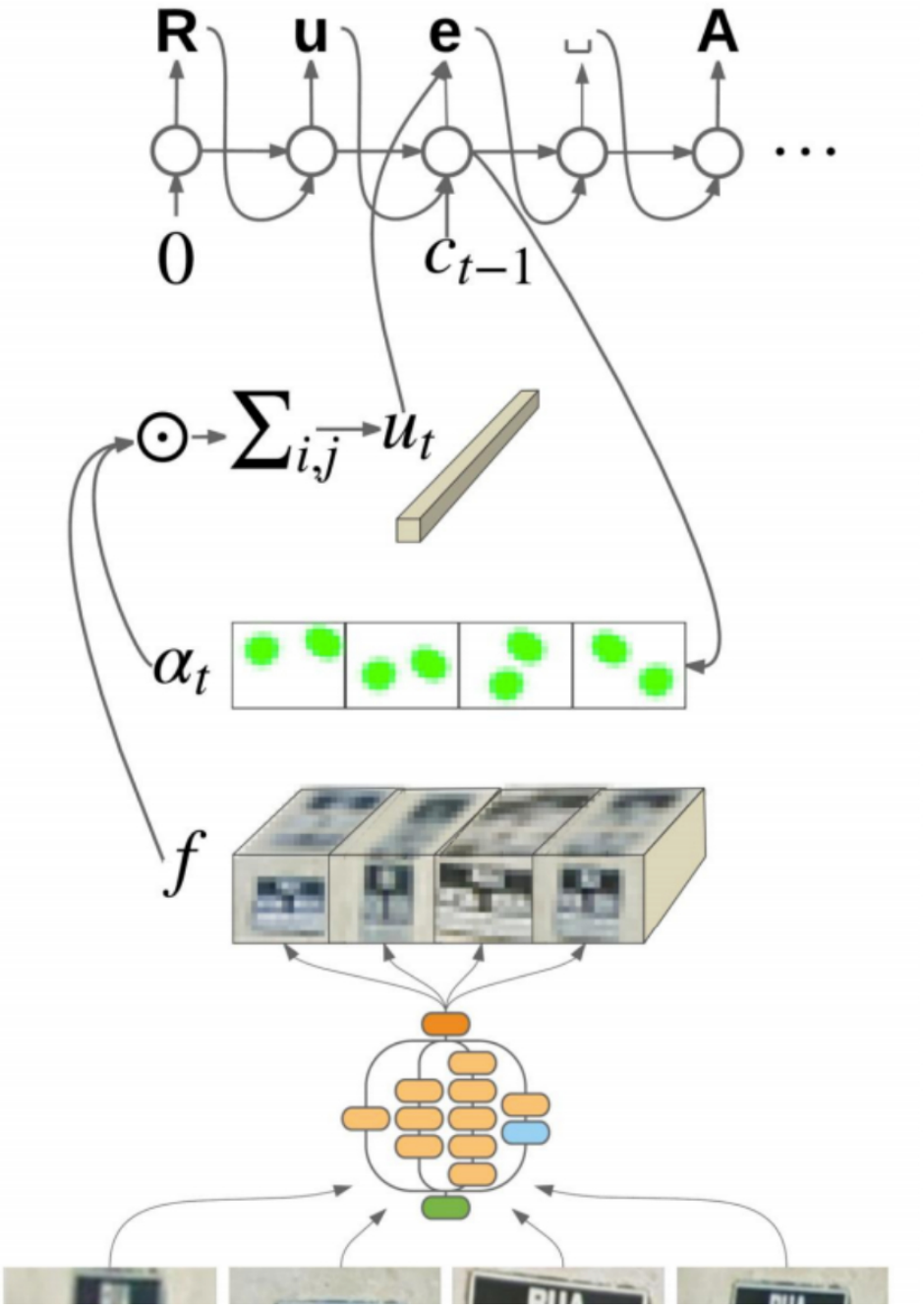
Image Source: https://arxiv.org/pdf/1707.08831.pdf

Results - The authors used the error gradients obtained by either calculating the cross-entropy loss or the CTC loss of the predictions and the textual labels. The Network achieved a 95.2% accuracy in identifying the text from the

images from the SVHN dataset. The same network was used on Robust Reading Datasets where the network performed at 90.3% accuracy.

Attention OCR

Attention OCR is a combination of both CNN and RNN with a novel attention mechanism. In this, we pass images which have different views through the same CNN feature extractor, and then concatenate the results into a single large feature map. First we use layers of convolutional networks to extract encoded image features. These extracted features are then encoded to strings and passed through a recurrent network for the attention mechanism to process. The attention mechanism used in the implementation is borrowed from the Seq2Seq machine translation model. We use this attention based decoder to finally predict the text in our image.



Source: https://arxiv.org/pdf/1704.03549.pdf

Results - Researched on Street View Imagery where it achieved 84.2% accuracy on the challenging French Street Name Signs (FSNS) dataset, significantly outperforming the previous state of the art (Smith'16), which achieved 72.46%. Normal CNNs gave up to 80.4% accuracy.

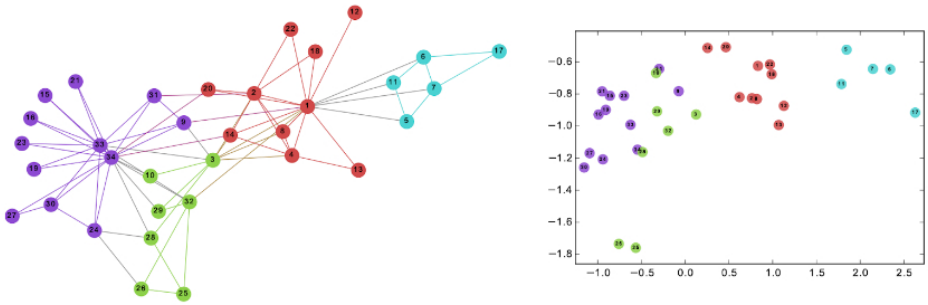
Why CNNs aren't enough

Convolutional Neural Networks work extremely well in tasks like image classification, video processing, speech recognition, and natural language understanding. The data in these tasks is typically represented in the Euclidean space. However, there is an increasing number of applications where data are generated from non-Euclidean domains and are represented as graphs with complex relationships and interdependency between objects.

This is where graph neural networks come into action.

Graph Neural Networks and Digitization

Graph Networks are a special type of neural networks that use graphs as input. If you are wondering what graphs are these are simple data structures that model objects (nodes) and relations (edges). The main idea is to encode the underlying graph-structured data using the topological relationships among the nodes of the graph, in order to incorporate graph-structured information in the data processing step.



Reference: DeepWalk: Online Learning of Social Representations

One of the most popular research based on Graph Networks is Deep Walk. The algorithm performs random walks to create node sequences. A skip-gram model is used to generate node embeddings which are then used for

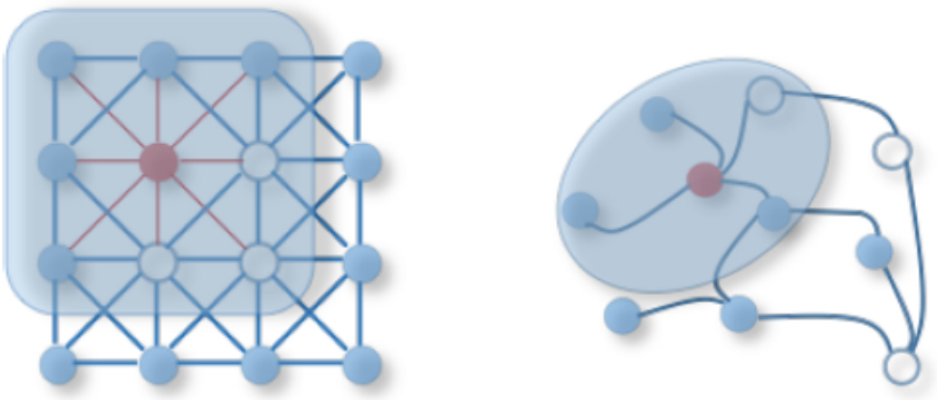
model is used to generate node embeddings which are then used for classifying these nodes. Above is an image of input and output of the deep network, Different colors in the graph indicates different labels in the input graph. We can see that in the output graph (embedding with 2 dimensions), nodes having the same labels are clustered together, while most nodes with different labels are separated properly.

Graph Convolutional Networks

In GCN, graph type input data is fed to it. In the feature extraction process, they use spectral and spatial approaches for performing convolution on graphs, with this, we can identify the coordinates of text in the ID cards or text documents with higher precision. One main advantage of using spatial features is that they do not require homogenous graph structures, meaning the inputs can be of different types. The use of normal pooling layers and subsampling layers is to highlight the features of the text. Once the feature extraction is complete, they use a classification network to identify the text found inside the coordinates and return the scores. With this, we can generate text and return the outputs as labels. You can learn more about graph networks by following this article and checking out the Github repository.

CNNs versus GCNs

Now, let's see the core difference between CNN and GCN. In the below images, the one on the left side represents a 2-dimensional convolution, where the pixel in an image is taken as a whole node and the neighbors are determined by the filter size (the neighbors can be a 3x3 filter, 4x4 filter, and so on). The 2D convolution takes the weighted average of pixel values of the red node along with its neighbors to highlight and extract the features from the images. On the other hand, in Graph Convolution, to find a hidden representation of the red node, we need to take the average value of the node features of the red node along with its neighbors.



2d convolution vs graph convolution - source

GCNs for visually rich documents

Texts in visually rich documents cannot be serialized into sequences without losing information. Classic information extraction models such as biLSTM-CRF based models were unable to take into account the visual features of the document image. This paper used graph convolutions to approach the problem in a way that would allow us to take into account the visual features as well.

Model architecture

Figure 3: Graph convolution of document graph. Convolution is defined on node-edge-node triplets (t_i, r_{ij}, t_j). Each layer produces new embeddings for both nodes and edges.

For GCNs, the inputs are loaded as graph signals. In the paper, they used graph convolution for information extraction from VRDs. Each text segment is comprised of the position of the segment and the text within it. Hence the position of the text components is determined by the four coordinates by using bounding boxes.

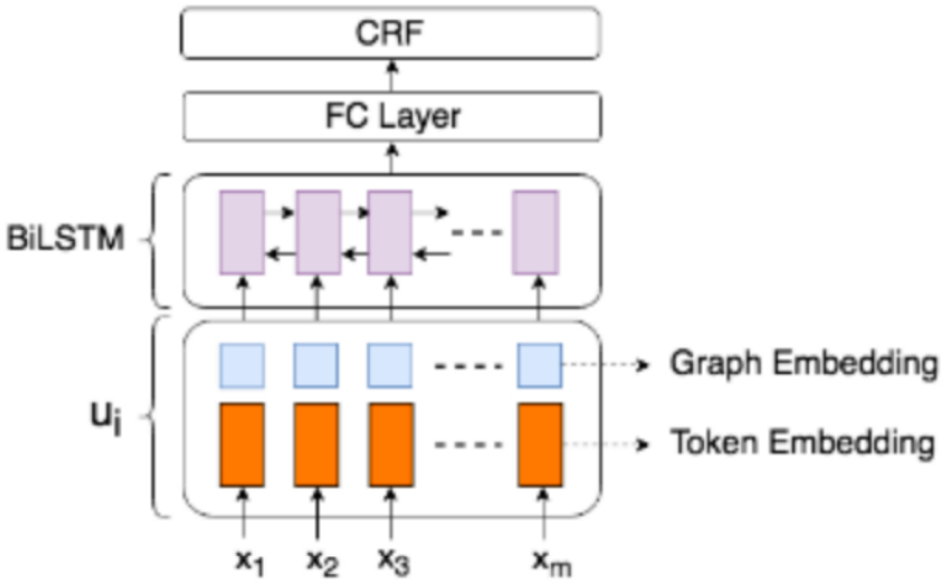


Figure 4: BiLSTM-CRF with graph embeddings.

They use a graph convolution based model to combine textual and visual information presented in VRDs. Graph embeddings are trained on all the text segments to get document vectors that are further combined with text embeddings (a bi-LSTM is used for this). The output is passed through a fully connected layer and a CRF layer to finally get our predictions.

Performance

The models are evaluated on their performance on two real world datasets - Value Added Tax Invoices (VATI) and International Purchase Receipts (IPR). The model is compared with two BiLSTM-CRF models. Baseline I applies BiLSTM-CRF to each individual sentence. BaselineII applies the IOB tagging model to the concatenated document. They also analyse their own model without visual features and text features extraction and without the attention mechanism and provide a comparison. The results are shown below.

Model	VATI	IPR
Baseline I	0.745	0.747
Baseline II	0.854	0.820
BiLSTM-CRF + GCN	0.873	0.836

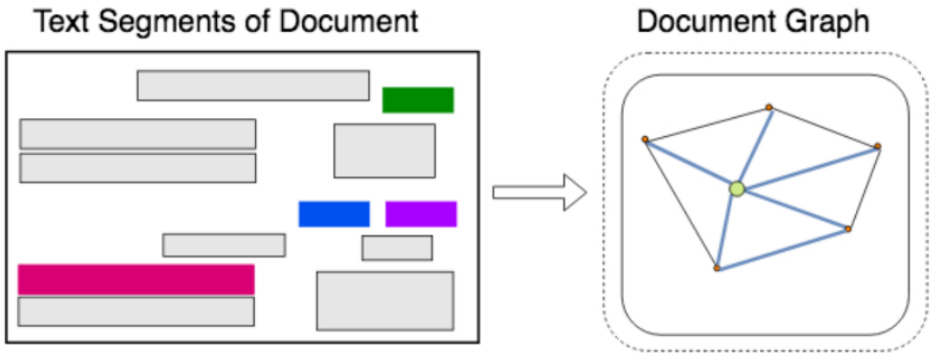
Table 1: F_1 score. Performance comparisons.

Configurations/Datasets	VATI	IPR
Full model	0.873	0.836
w/o vis. features	0.808	0.775
w/o text features	0.871	0.817
w/o attention	0.872	0.821

Table 3: F_1 score. Ablation studies of individual component of graph convolution.

GCNs for ID card digitization

Below is how the inputs are loaded as document graphs, and each node is connected to one another. In the graph, the nodes represent text segments (Usually OCR) and the edges represent the visual dependencies, such as relative shapes and distance, between the two nodes.



Now consider the problem of ID card information extraction, imagine all the nodes in the graph (which we'll be using as input) are associated with a label (ID card fields) and we want to predict the label of the nodes. Here, our nodes are every word in the document, their feature vectors are their word vectors concatenated with bounding box information and the labels are what field they represent - name, address, etc. With this input, the network learns node respresentations that understands the information about its neighborhood nodes. These graph embeddings are passed through a biLSTM followed by a fully connected layer to finally get the different fields. the biLSTM outputs can be used to sequentially generate the OCR text and it's location by passing them through a decoder architecture.

ID card digitization with Nanonets

The Nanonets OCR API allows you to build OCR models with ease. You can upload your data, annotate it, set the model to train and wait for getting predictions through a browser based UI without writing a single line of code, worrying about GPUs or finding the right architectures for your deep learning models.

SEX	Male
CLASS	

ID card digitization in 15 mins

You can upload your own data and train a model, acquire the JSON responses of each prediction to integrate it with your own systems and build machine learning powered apps built on state of the art algorithms and a strong infrastructure.

Here's how -

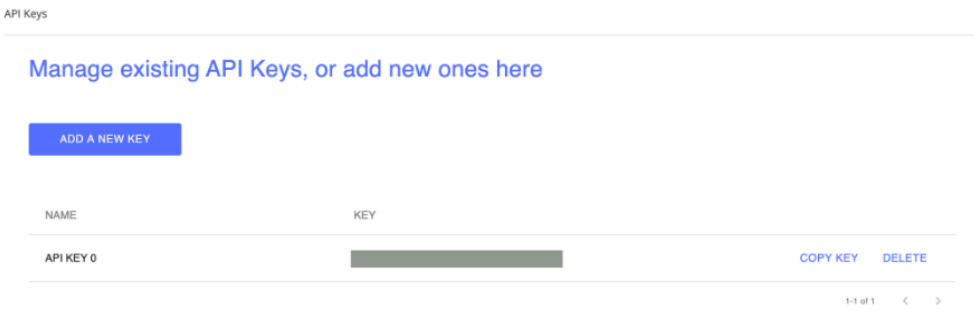
Using the GUI: https://app.nanonets.com/

You can also use the Nanonets-OCR API by following the steps below:

Step 1: Clone the Repo, Install dependencies (Repo Link)

```
git clone https://github.com/NanoNets/nanonets-id-card-digitization.git
cd nanonets-id-card-digitization
sudo pip install nanonets
```

Step 2: Get your free API Key
Get your free API Key from http://app.nanonets.com/#/keys



Step 3: Set the API key as an Environment Variable

```
export NANONETS_API_KEY=YOUR_API_KEY_GOES_HERE
```

Step 4: Upload Images For Training
The training data is found in `images` (image files) and `annotations` (annotations for the image files)

```
python ./code/training.py
```

Note: This generates a MODEL_ID that you need for the next step

Step 5: Add Model Id as Environment Variable

```
export NANONETS_MODEL_ID=YOUR_MODEL_ID
```

Note: you will get YOUR_MODEL_ID from the previous step

Step 6: Get Model State

The model takes ~2 hours to train. You will get an email once the model is trained. In the meanwhile you check the state of the model

```
python ./code/model-state.py
```

Step 7: Make Prediction

Once the model is trained. You can make predictions using the model

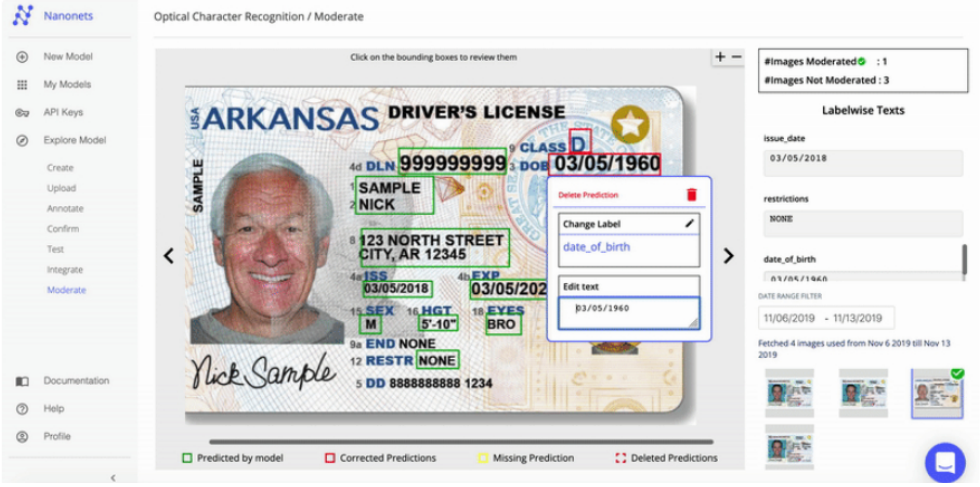
```
python ./code/prediction.py PATH_TO_YOUR_IMAGE.jpg
```

Sample Usage:

```
python ./code/prediction.py ./images/111.jpg
```

Nanonets and humans in the loop

The 'Moderate' screen aids the correction and entry processes and reduce the manual reviewer's workload by almost 90% and reduce the costs by 50% for the organisation.



Features include

1. Track predictions which are correct
2. Track which ones are wrong
3. Make corrections to the inaccurate ones
4. Delete the ones that are wrong
5. Fill in the missing predictions
6. Filter images with date ranges
7. Get counts of moderated images against the ones not moderated

All the fields are structured into an easy to use GUI which allows the user to take advantage of the OCR technology and assist in making it better as they go, without having to type any code or understand how the technology works.

Summary

In this article, we've discussed in brief about information extracting from the ID cards and digitizing them. We've reviewed all the popular techniques with their advantages and disadvantages.

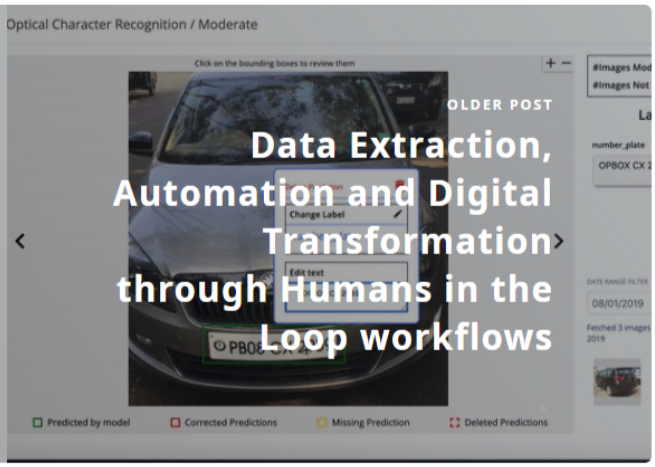
We've seen how neural networks work, and how convolutions neural networks are used to identify the features and patterns in images. We discuss different methods that have been used for OCR tasks until now and provide a comparison for which techniques work well. Later, we've discussed one of the popular techniques for information extraction, GCNs. They achieved the state of the art performance in the information extraction process from the scanned documents.

Lastly, ID Card Digitisation reduces a lot of time and human efforts in several organizations and business models. Using Deep Learning, we can automate this problem and deploy solutions in real-time across different applications. These models not only reduce costs and efforts but also are scalable and high performing. However, the art of building a state of the art performance model requires a lot of research and experimentation. This article helps you understand how where to start and gives you a brief about all the popular techniques.

M↓ MARKDOWN

ADD COMMENT

Powered by **Commento**



PRODUCTS

- Object Detection
- Image Classification

SOLUTIONS

- NSFW
- Drones
- E-commerce
- Inspection
- Multi Label Classification
- OCR API
- Hygiene & Safety Compliance
- Insurance

CASE STUDIES

- Counting Cars
- Solar Panel faults
- Windmill faults
- NSFW Content Moderation
- Furniture Research & Recommendation

COMPANY

- About Us
- Blog

CONTACT

156 2nd Street, San Francisco, CA 94105, USA

+1 650 382 8676

info@nanonets.com