Ellie mae take home assessment

1. 20 news group classification

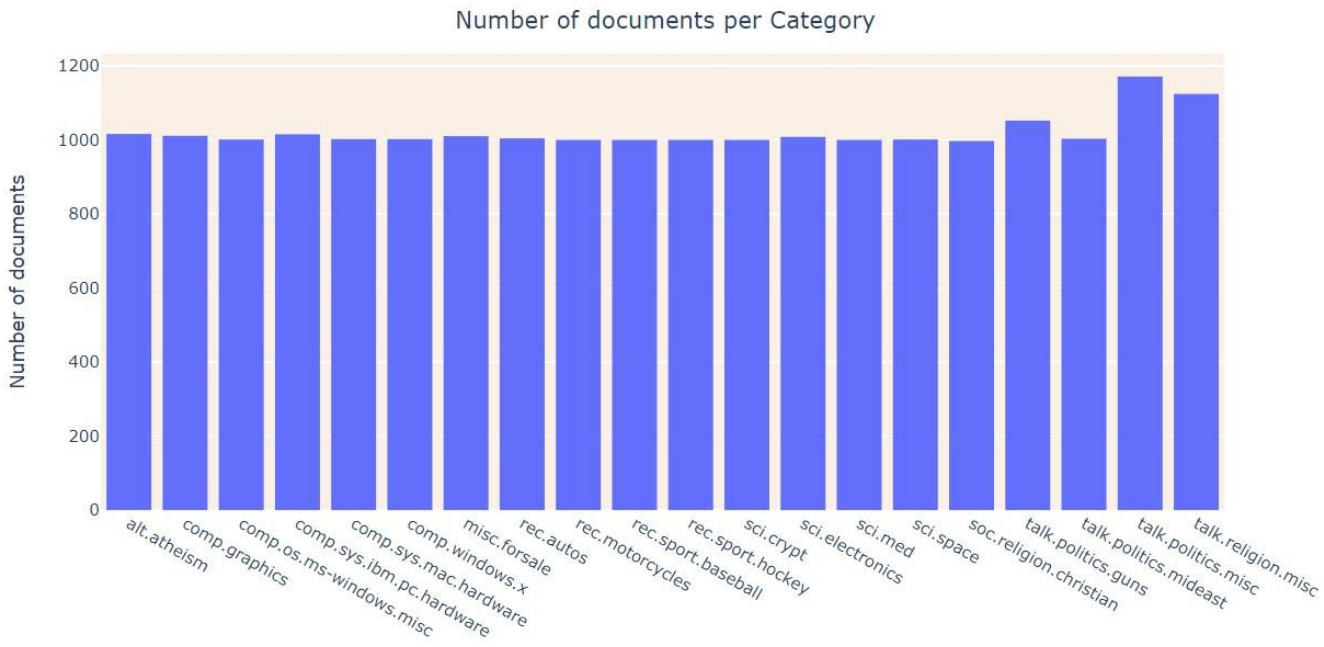
a. Overview of the data and Exploratory data analysis:

There are 20,417 news group documents and the data is organized into 20 different newsgroups, each corresponding to a different topic.

EDA:

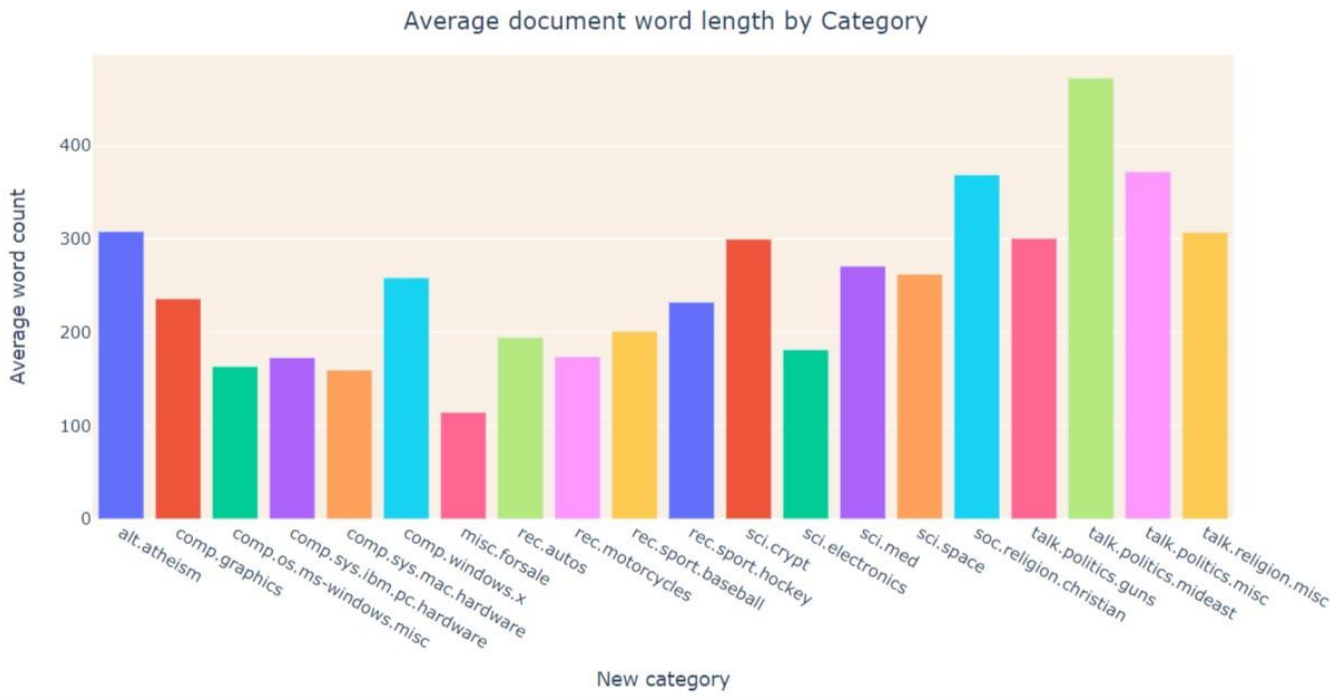
I have done some exploratory data analysis which helped me to get an overview of the data and to formulate the model. There aren't any empty documents. However, data is a bit messy.

First let's see distribution of the data:



As we can see from above figure, there are over 1000 documents for each new category. Also, the data is evenly distributed without any imbalance.

Average document length by category:



Category misc.forsale has minimum number of words per document: 114

Categories talk.politics.mideast has maximum number of words per document: 472

This helped in choosing the sequence length or embedding length while building machine learning models.

Most frequent words in each news group:

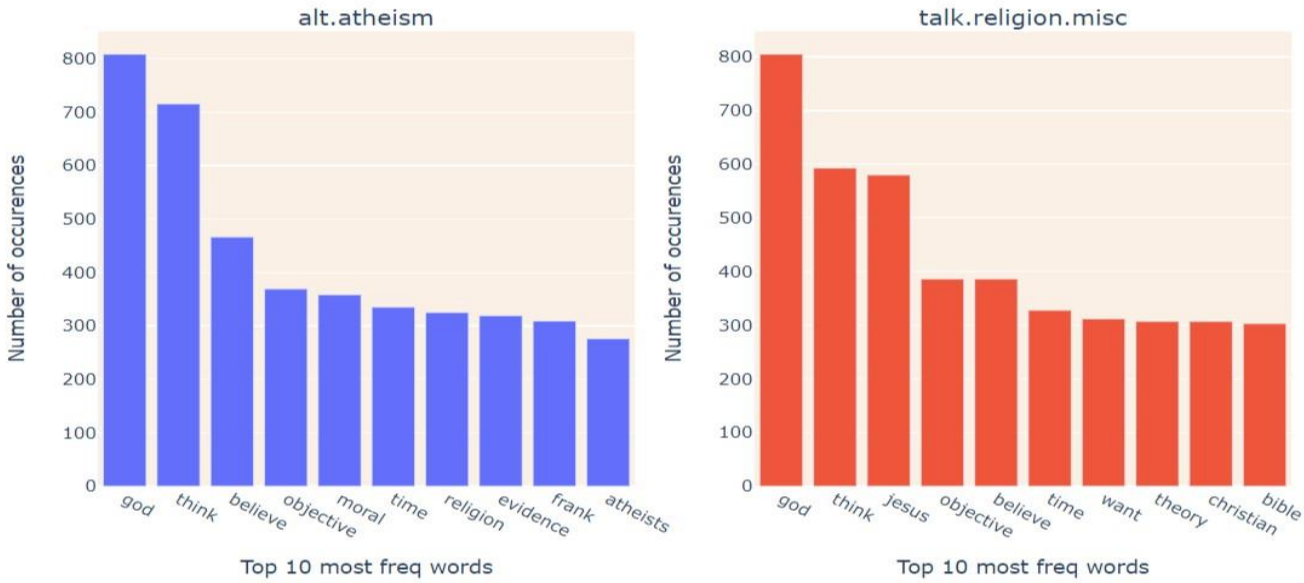


Fig.3. Most frequent words of atheism and religion.misc categories

b. Model selection:

Model building- As we have to classify each document to one of 20 news group categories, I choose this as multiclass classification problem.

Evaluation metrics- I used micro f1 score and accuracy scores as base evaluation metrics.

Train test split- As there is no explicit testing data provided, I used train_test_split to create separate train and test data sets. Throughout modeling, I used train_test_split ratio of 0.2 and random_state=9 to obtain consistent results. There are 16333 train samples and 4084 test samples.

Model selection- As per requirements, I have to train one traditional ml model and one neural network model.

For traditional model, I tried three variants: a) bag of words as tokenizer and logistic regression b) TF-IDF tokenizer and Logistic regression c) TF-IDF tokenizer and SVM.

I choose distil-bert as neural network variant and compared both models. One primary reason for choosing distil-bert is contextual embedding which I will discuss later.

c. Data preprocessing:

As we are dealing with text data, I followed few steps to clean the data. I removed special characters, non-ascii characters, extra white spaces and numbers. I converted text to lower case for proper tokenization. For svm, I removed stop words and lemmatized words. I haven't removed stop words for bert as they are necessary for bert model building.

d. Traditional machine learning models:

i. Bag of words and Logistic regression:

The bag-of-words model is a simplified representation of the raw data. In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words. Bag-of-words representations discard grammar, order, and structure in the text, but track occurrences. To use this model, Sklearn offers a `CountVectorizer` class which returns a sparse matrix and basically does the same specified before, but which has some configurable options like max_features to be used.

I used logistic regression with OneVersusRest configuration as we are dealing with multi class problem.

ii. TF-IDF and Logistic regression:

In second scenario, I used term frequency-inverse document frequency (tf-idf) in place of bag-of-words model. tf-idf tells us which words are most discriminating between documents. Words that occur a lot in one document but don't occur in many documents contain a great deal of discriminating power.

- This weight is a statistical measure used to evaluate how important a word is to a document in a collection (aka corpus).

- The importance increases with the number of times a word appears in the document but is offset by the frequency of the word in the corpus.

The inverse document frequency is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word plus one, obtained by dividing the total number of documents by the number of documents containing the term plus one, and then taking the logarithm of that quotient.

This enhances terms that are highly specific of a particular document, while suppressing terms that are common to most documents.

Like previous model, I tested the transformed tf-idf features on logistic regression.

ii. TF-IDF and SVM:

In third scenario, I used features extracted from tf-idf on SVM. SVM tries to classify a set of points by maximizing the margin between the classes using support vectors. I performed hyper parameter tuning on SVC.

Parameters tuned: kernels (linear, rbf, poly), gamma and c parameters.

Results:

model	F1 score	accuracy
CBOW +Logistic regression	0.83	0.82
Tf-idf + Logistic regression	0.79	0.79
Tf-idf + SVM	0.74	0.73

As we can see from above results, bag of words + logistic regression is performing better than other 2 models with an F1 score of 0.83 and accuracy of 0.82. One primary reason is as many words are common between different news groups, tf-idf is getting impacted. For example, as shown in figure 3, most frequent words of ‘alt.atheism’ and ‘talk.religion.misc’ are almost same. So it is impacting inverse document frequency score(idf score) as these words are present in multiple news categories and categories and thus impacting model performance. Also logistic regression is performing better than SVM as the feature space is huge and SVM with non-linear kernel(rbf) is suffering with curse of dimensionality.

e. Distilbert:

Few reasons for choosing distilbert:

a. There are less training samples. So, it's hard for normal ml models to capture the essence of the data with such less examples. On other hand bert variants help as one can use transfer learning to build decent models without many training examples.

b. Context based learning. Often documents are context based. For example, lets take 2 sentences: 'I believe in god' and 'I don't believe in god'. Using non contextual embeddings like cbow or tfidf, these two sentences are closely related and thus the models get confused. In this example with word "don't", whole meaning of message has changed. Bert variants are better when compared to other ml models for such examples due to their contextual understanding using masked language modeling and multi-headed attention mechanisms.

c. Word piece tokenizer: There are multiple words which carry similar meaning. For example 'ok and okay' or 'annoyed and annoying'. Many tokenizers considers these as different words where as bert breaks down words using word piece tokenizer something like 'annoy + ed' and 'annoy + ing' and there is high possibility that when a new word appears, bert can associate it with closest word embedding.

i.) Baseline model

First, I built a baseline distilbert model to see how it performs on the data. For baseline model, I used pretrained distil-bert model provided by hugging face and tried to classify models without any architecture change. Loss converged well and training accuracy is reached 95% whereas on test data, F1 score and accuracy are 0.83. However one primary thing I observed is all the models are getting confused between alt.atheism and talk.religion.misc. F1 score on category atheism is 0.63 and F1 score on religion is 0.52. Often models are classifying atheism as religion and viceveversa.

ii.) DocBert model

To improve the shortcomings of previous models, I used DocBert architecture. I am inspired by DocBert BERT for Document Classification architecture developed by Adhikari et al., 2019 which tries to distil knowledge from bert to LSTMs which are proven to be effective due to long text format of documents. This architecture tries to remove ambiguity between closely related sentences using combination of Bert and LSTMs. DocBert is current state of the art solution for document classification on Multiple datasets like reuters dataset. So I implemented DocBert architecture and tweaked it on 20 Newsgroup dataset. Using DocBert, F1 jumped from 0.83 to 0.87. Then I started to tune the docbert model.

Hyperparameter tuning:

I turned on to hyperparameter tuning to identify best parameters.

Parameters tuned:

Epochs: 3 to 20.

Batch size: 4, 8, 16.

Learning rate: 1e-5, 2e-5 and 3e-5 for Adam optimizer.

Maximum sequence length: 128,256 and 512.

LSTM components like number of hidden dimensions, embedding dimensions for docbert model.

Dropout

After parameter tuning, the f1 score improved to 0.9 and accuracy increased to 88.6%.

e. Results and thoughts:

Model	F1 score	Accuracy
CBOW +Logistic regression	0.83	0.82
Tf-idf + Logistic regression	0.79	0.79
Tf-idf + SVM	0.74	0.73
Distilbert baseline	0.834	0.831
Docbert baseline	0.87	0.85
Docbert with param tuning	**0.9**	**0.886**

Final results.

As we can see from above the F1 score is surprisingly good for cbow + logistic regression model (0.83) when compared to SVM variants (0.74). One primary thing I observed is CBOW is helping the model better than tf-idf as many words are repeated in multiple documents across different categories. However, performance is not improving above 0.83 for traditional machine learning models. One primary reason is using CBOW and tf-idf looses semantic meaning of the doc and thus they couldn't classify documents properly. So, I moved to distil-bert. Baseline model is having performance close to cbow+logistic regression. I then used Docbert which combines embeddings from distil-bert with LSTMs and finetuned it. It has huge improvement in terms of performance. The F1 score jumped to 0.9.

Space for improvement:

a. I observed models are confused more between alt.atheism and misc.religion categories. F1 score for religion category is 0.57 using logistic regression, 0.50 using SVM and 0.62 using Docbert. Using traditional machine learning models with cbow and tfidf, we can't distinguish two closely related categories. However, using distil-bert variants, if we have few more training examples, we can improve F1 score of closely related categories by huge margin.

b. Also using bert instead of distil-bert will improve model performance a bit. It is more like performance vs speed tradeoff. Training speed and inference speed will be more for distil-bert as it is over 65% faster than bert while performance is typically 95% of bert.

c. I capped maximum word length to 512 due to resource constraints. I used google colab for training distil-bert and it provides gpu ram of 8gb. I tried to increase maximum sequence length, but it overflowed the memory. Increasing the sequence length will improve performance.

d. If more resources and time are available, I might have tried Unroll Bert architecture (https://andriymulyar.com/blog/bert-document-classification) which seems pretty interesting. One major issue while dealing with documents is its length or number of words present. For large documents, we can't cap the word length to 512 or 1024. So Unroll bert breaks down document to sentences and feeds each sentence to bert. Finally, it uses sequence to sequence modeling using LSTMs and combines all sentences to classify the document. Using this architecture, we can use full document for classification unlike other architectures.

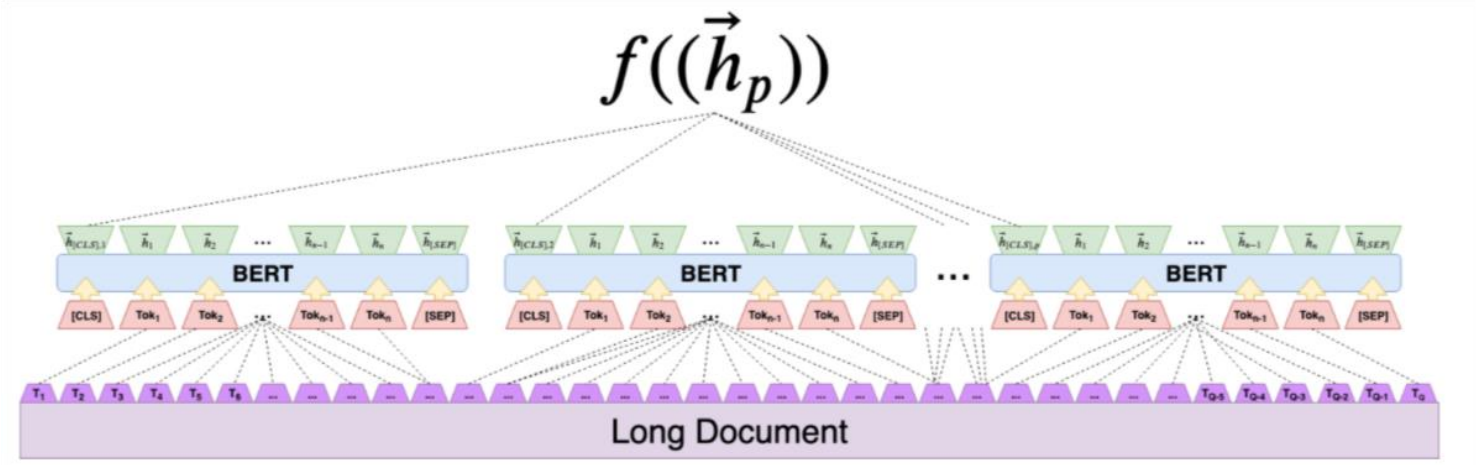


Figure 2: Document Bert. A function f parameterized by a seq2seq neural network condenses a sequence of BERT hidden states into a global document context vector. For instance, f could be an LSTM cell.

2. Information extraction from documents and images

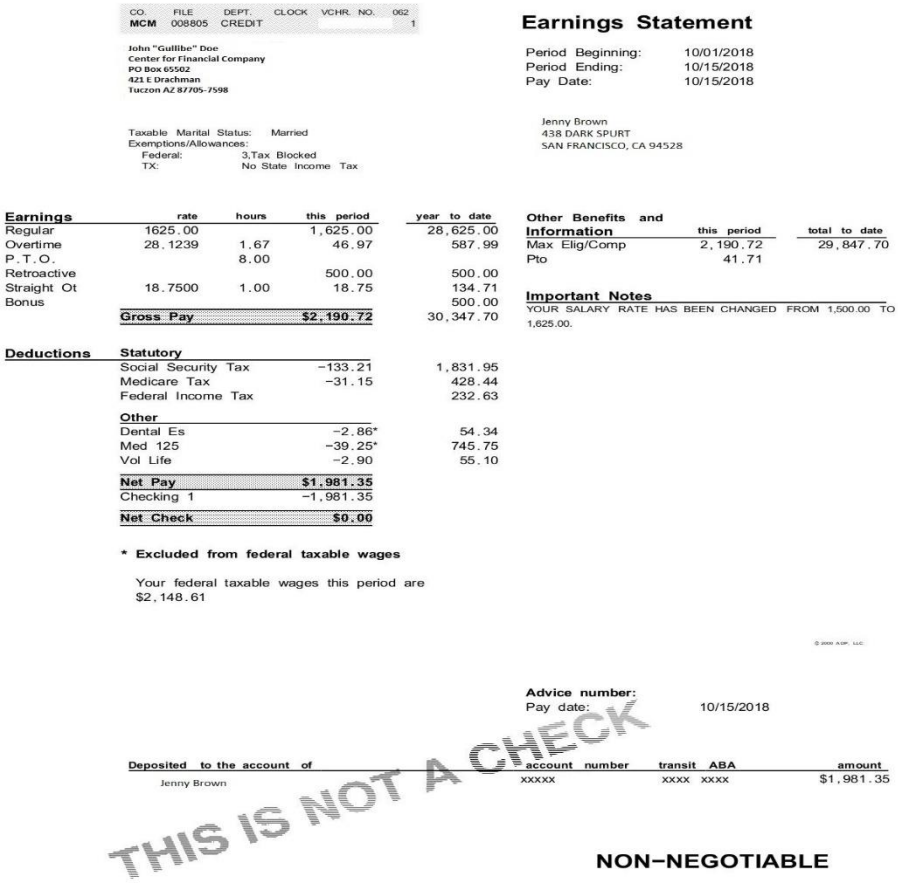
This is one of the most interesting and challenging problems which comes at intersection of computer vision and NLP space. It is still an open research area and many companies are working on it. There are multiple ways to approach this problem.

1. Using existing solutions like Amazon textract as baseline model and building models on top of the results.
2. Build end to end machine learning models using architectures like DeepDeSRT, Tablenet etc.

One major thing using traditional OCR tools like pdfbox, epesoft, tesseract is they often loose spatial information of documents.

Using Amazon textract and building models on top of it:

Textract extracts raw text(OCR), clubbed words, key-value pairs, bounding boxes with confidence and tables from images and pdfs. I used a sample paystub provided to see how data is extracted from Textract Paystub-Co-Borrower_var0014(6).pdf1:



CO. FILE DEPT. CLOCK VCHR. NO. 062
MCM 008805 CREDIT 1

John "Gullibe" Doe
Center for Financial Company
PO Box 65502
421 E Drachman
Tucson AZ 87705-7598

Taxable Marital Status: Married
Exemptions/Allowances:
Federal: 3,Tax Blocked
TX: No State Income Tax

Earnings Statement

Period Beginning: 10/01/2018
Period Ending: 10/15/2018
Pay Date: 10/15/2018

Jenny Brown
438 DARK SPURT
SAN FRANCISCO, CA 94528

Earnings	rate	hours	this period	year to date
Regular	1625.00		1,625.00	28,625.00
Overtime	28.1239	1.67	46.97	587.99
P.T.O.		8.00		
Retroactive			500.00	500.00
Straight Ot	18.7500	1.00	18.75	134.71
Bonus				500.00
Gross Pay			$2,190.72	30,347.70

Other Benefits and Information	this period	total to date
Max Elig/Comp	2,190.72	29,847.70
Pto	41.71	

Important Notes
YOUR SALARY RATE HAS BEEN CHANGED FROM 1,500.00 TO 1,625.00.

Deductions	Statutory		
	Social Security Tax	-133.21	1,831.95
	Medicare Tax	-31.15	428.44
	Federal Income Tax		232.63
	Other		
	Dental Es	-2.86*	54.34
	Med 125	-39.25*	745.75
	Vol Life	-2.90	55.10
	Net Pay	$1,981.35	
	Checking 1	-1,981.35	
	Net Check	$0.00	

* Excluded from federal taxable wages

Your federal taxable wages this period are $2,148.61

Advice number:
Pay date: 10/15/2018

Deposited to the account of	account number	transit ABA	amount
Jenny Brown	xxxxx	xxxx xxxx	$1,981.35

THIS IS NOT A CHECK

NON-NEGOTIABLE

For the above sample key value pairs extracted by textract:

Period Beginning:10/01/2018, Period Ending:10/15/2018, Net pay: $1981.35 etc. Depending on the problem key-value pairs will be a good starting point for information extraction.

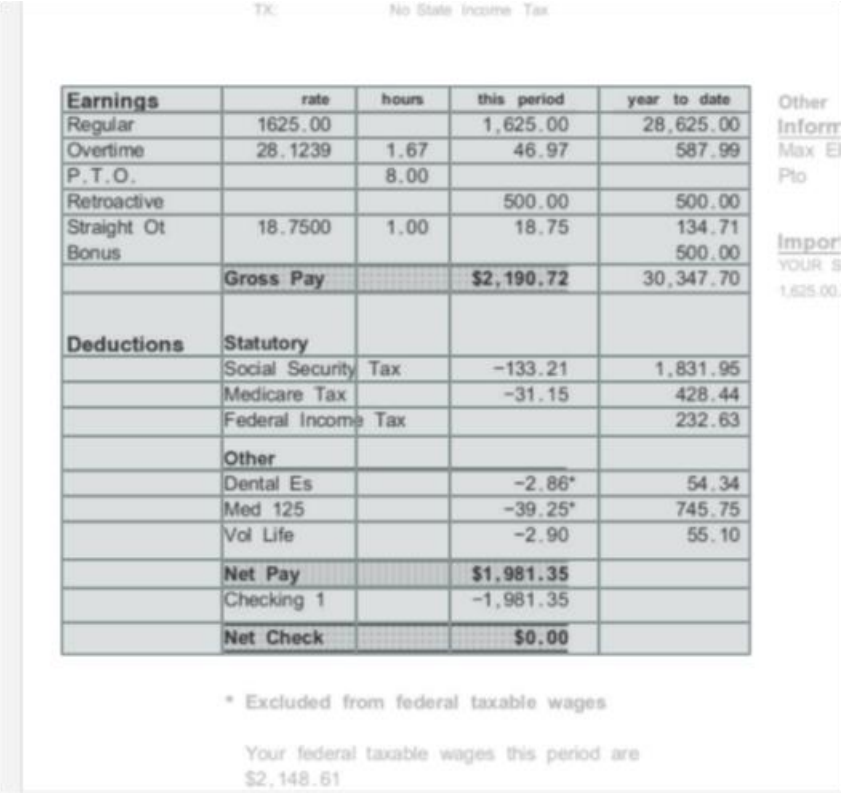
We can extract human names, company names and all the addresses using NER model. Using the raw text extracted from textract, NER models can be built with good accuracy.

Textract also provides bounding boxes with confidence:


```
{
  "BlockType": "LINE",
  "Confidence": 99.94651794433594,
  "Text": "Federal Income Tax",
  "Geometry": {
    "BoundingBox": {
      "Width": 0.13977457582950592,
      "Height": 0.009956118650734425,
      "Left": 0.15425215661525726,
      "Top": 0.42344385385513306
    },
    "Polygon": [
      {
        "X": 0.15425215661525726,
        "Y": 0.42344385385513306
      },
      {
        "X": 0.2940267324447632,
        "Y": 0.42344385385513306
      },
      {
        "X": 0.2940267324447632,
        "Y": 0.4333999752998352
      },
      {
        "X": 0.15425215661525726,
        "Y": 0.4333999752998352
      }
    ]
  },
  "Id": "19ce6a33-b212-4701-a47a-5bb0d7b9f695",
  "Relationships": [
    {
      "Type": "CHILD",
      "Ids": [
        "da682cba-090e-46fa-aaf4-5a0ee1404242",
        "d8180285-88aa-4e4b-afac-e2e779f91385",
        "1deb843c-b087-420c-a0d4-d8e22f9ce17c"
```


This reduces the necessity of manual keyers to draw bounding boxes and typically saves so much time and money. Also one can build **Spatial KDtree** to create spatial 2d map of all coordinates and identify key values pairs pretty effectively. Textract provides parent and child relations which makes it effective to extract key-value pairs.

Also, Textract extracts tables from which we can extract more information. Sample table detected for above example:



TX No State Income Tax

Earnings	rate	hours	this period	year to date
Regular	1625.00		1,625.00	28,625.00
Overtime	28.1239	1.67	46.97	587.99
P.T.O.		8.00		
Retroactive			500.00	500.00
Straight Ot	18.7500	1.00	18.75	134.71
Bonus				500.00
	Gross Pay		**$2,190.72**	30,347.70
Deductions	**Statutory**			
	Social Security Tax		-133.21	1,831.95
	Medicare Tax		-31.15	428.44
	Federal Income Tax			232.63
	Other			
	Dental Es		-2.86*	54.34
	Med 125		-39.25*	745.75
	Vol Life		-2.90	55.10
	Net Pay		**$1,981.35**	
	Checking 1		-1,981.35	
	Net Check		**$0.00**	

* Excluded from federal taxable wages

Your federal taxable wages this period are $2,148.61

Other Inform Max E Pto

Impor YOUR S 1,625.00

So using solutions like Textract and building models like NER, spatial KDtree, table extraction on top of it will be a good solution for key-value extraction. Also, the bounding boxes will save manual labor by removing need of keying.

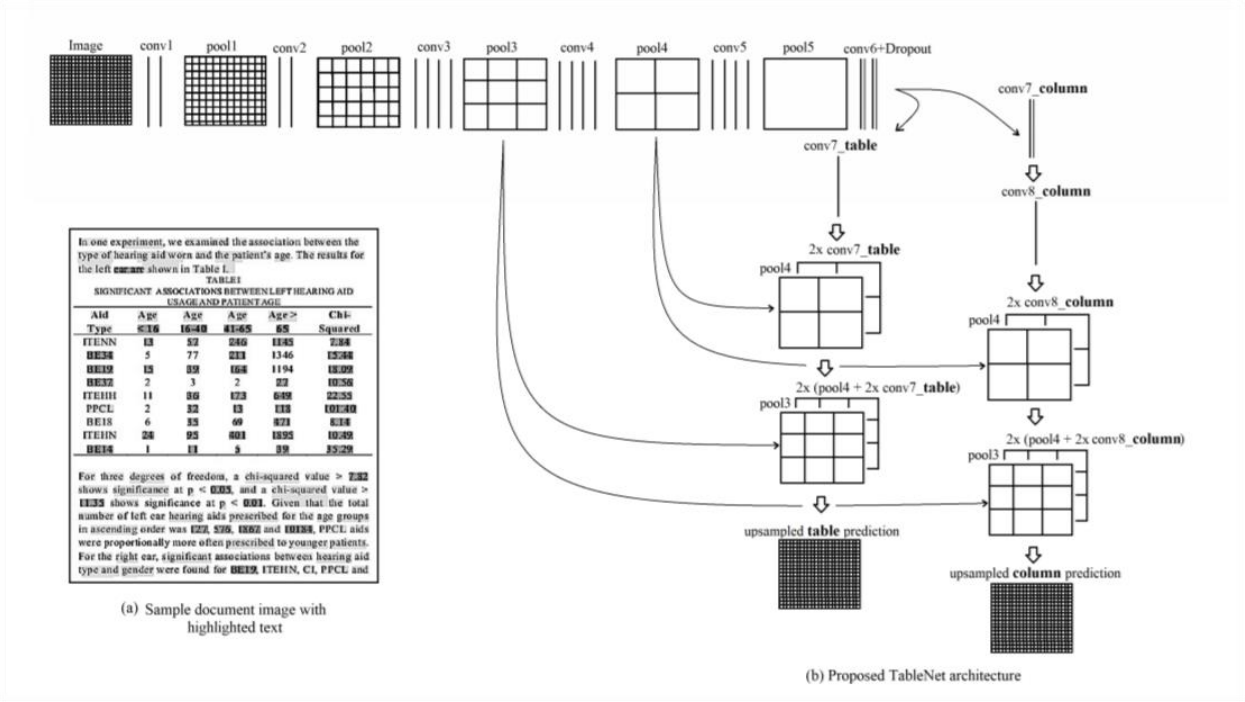
DeepDeSRT AND Tablenet architectures:

Second approach is building end to end machine learning models using architectures like DeepDeSRT. Majority of end to end solutions revolve around table detections and structure recognition. One major challenge in key value extraction using object detection models is that text is closely related. For example in above image, Social security tax, medicare tax and federal income tax are very similar and also they are located in close proximity. So typical deep learning methods involves more table detection and going to more granular level.

1. Identify tables in image or document.
2. Identify rows and columns in the table detected.

The number of tables per document will typically be less. At max, there will be 5 or 6 in normal scenarios. However for each table detected, there can be many rows and columns and their structures change wildly.

To solve this, Tablenet proposes a base network that is initialized with pre-trained VGG-19 features which acts as an Encoder. It is followed by two decoders: One for Segmentation of the table region and second decoder for Segmentation of the columns within a table region. Subsequently, rule based row extraction is employed to extract data in individual table cells. The decoders are built on semantic segmentation using FCN(fully convolution networks) with skip architectures. TableNet has pretrained models available. So, one can use this architecture as baseline model and finetune for sub tasks like for paystubs.



In one experiment, we examined the association between the type of hearing aid worn and the patient's age. The results for the left ear are shown in Table I.

TABLE I
SIGNIFICANT ASSOCIATIONS BETWEEN LEFT HEARING AID USAGE AND PATIENT AGE

Aid Type	Age ≤16	Age 16-40	Age 41-65	Age > 65	Chi-Squared
ITENN	13	52	940	1145	7284
BE34	5	77	813	1346	15548
BE19	15	89	864	1194	18109
BE32	2	3	2	23	10856
ITEHH	11	86	173	649	22355
PPCL	2	32	13	118	101840
BE18	6	35	69	871	814
ITEHN	24	95	801	1895	10849
BE14	1	11	5	39	35729

For three degrees of freedom, a chi-squared value > 7.82 shows significance at p < 0.05, and a chi-squared value > 11.35 shows significance at p < 0.01. Given that the total number of left ear hearing aids prescribed for the age groups in ascending order was 122, 570, 4362 and 10188, PPCL aids were proportionally more often prescribed to younger patients. For the right ear, significant associations between hearing aid type and gender were found for BE19, ITEHN, CI, PPCL and

(a) Sample document image with highlighted text

(b) Proposed TableNet architecture

Also, often the quality of images or documents maybe poor. If the quality is poor, we have use image processing techniques to enhance image quality.

End notes:

There isn't a single best solution for this problem yet. We can use tools like Textract which provide key-value pairs, clubbed words, raw text, bounding boxes and tables as baseline. Or we can use architectures like Tablenet as baseline. On top of these, we have to use an Ensemble of machine learning models to further extract important information. Different models like NER, spatial kd tree mapping, table-based extraction can be used to get key-value pairs from documents.

Tablenet: Tablenet paper

DeepDeSRT: DeepDeSRT

Few companies working on this problem:

1. Hyperscience: https://hyperscience.com/
2. Amazon textract: https://aws.amazon.com/textract/
3. Cinnamon ai: https://cinnamon.is/en/