Machine Learning to Improve a Document Pipeline

Scott A. Wallace, Bhadresh Patel, Landon Kryger
Washington State University Vancouver
Vancouver, WA

Susan Seley
Data Data Inc.
Vancouver, WA

Abstract

We describe a collaborative project between our research group and a small west-coast business to apply machine learning techniques to a document processing task. This experience suggests two key points: (1) even as machine learning and artificial intelligence matures, there are many business applications that have not yet exploited these techniques; and (2) academically well-established machine learning techniques have much to offer both in terms of flexibility and economic benefit.

Introduction

A few years ago, our research group engaged in a collaborative project with a small west-coast business (Data Data, Inc.) whose mainstay product is a subscription service of information procured from public record documents across a large range of counties with varying populations. The business's document processing involved four main steps: (1) purchasing records from local municipalities; (2) categorizing documents; 3) extracting data from the documents; and 4) entering the extracted information into a database. Of great surprise to us at the time was the fact that much of this process (steps 2–4) were implemented by a third party using an outsourced labor pool. Large quantities of these scanned documents were being sent to India first to be categorized and then to have their data extracted and entered into a database. We recognized that well-established machine learning techniques could be applied to automate much of this process and our collaboration began.

Our first task was to examine the potential impact of automating this document pipeline. Because transitioning from a process based on human labor to one based on machine learning required a significant leap of faith from the management, we wanted to ensure that we looked at a spectrum of approaches that would range from minimal automation to full automation. As it turned out, the process description itself provided a nice scaffolding for this. In particular, we could easily envision a partially automated process that performed categorization (step 2) while leaving data extraction (step 3) to humans; similarly we anticipated that some data extraction tasks may be easier than others and this might allow us to scaffold step 3 further.

At the time, the Data Data's outsourced documents were being manually examined to identify 20 fields that were then entered by hand into a database. The business was charged a fixed fee per field and per document for the data entry. Interestingly, many of these database entries were in fact irrelevant to them—as only a small fraction of the documents were important for their products. However, because the records were typically purchased in bulk by the day, there was no easy way to identify each particular document's type *a priori*. Instead, someone needed to examine the scanned image to determine if they were relevant. The least expensive way to perform this task was to send all the documents abroad.

After some discussion, we arrived at the following series of potential automation tasks:

2-Class Categorization This is the simplest problem, in which we determine whether a particular document is relevant to the business prior to sending it off for manual information extraction. Thus, all the manual work would proceed as normal, but with a reduced document set that contained only useful items.

Multi-class Categorization Four of the twenty database fields associated with each document contained information derived from the document's specific type. Public record document types run the gamut and include "liens", "deeds", "deeds of trust", and "marriage licenses" among many others. Data Data was focused on data pertaining to property transactions and was interested in only 25 of the roughly 120 major document types. By correctly identifying the category to which each relevant document belonged, entries for four data fields could be automatically filled.

Stamp Classification Auditor and Recorder stamps contain critical information such as the date the document was recorded, the document number, document type information, and the name of the document recorder. In addition, by locating and interpreting these stamps, it is possible to identify documents that pass through the municipal recording system multiple times. Such re-recordings occur quite frequently but must be ignored for the business's report (only the initial recording is salient). Thus, by correctly reading data from stamps that appear on the document, some documents can be deemed irrelevant

(because they are re-recordings), and relevant documents may have four additional data fields automatically filled. Although these stamps can be knowledge rich, as we will discuss, their use does have limitations. Most critically, the stamps are county-specific and the document type information they encode is based on a county-specific taxonomy that does not always help isolate the document types in which we are interested. This means that "reading the label" will generally not solve even the 2-class categorization problem.

Information Extraction Associated with each document class is a pre-specified list of fields that must be extracted to generate the document's database entry. This list covers nine of the remaining twelve data fields and include such items as the location of the property, the document's recording number, date, and the names of the owners. However, accurately acquiring this information is non-trivial. Compared with classification that typically relies on optical character recognition performing reasonably well *in aggregate*, information extraction targets much smaller and more specific pieces of text and therefore relies much more heavily on the accuracy of the OCR software.

Implicit Information Inference The final three data fields pose the greatest challenge as they are not necessarily explicitly available in the document itself. For example, certain documents are associated with a loan although there is not necessarily any single piece of text that identifies the specific loan type. One loan type, *construction*, can be inferred if the deed has a special rider attached to it. Another type, *tied with*, can be inferred if two documents, a deed and deed of trust, have the same order number (identified by a title company's stamp) and are processed by the municipality in a specific chronological order. In either case, the data for these final fields must be inferred from a combination of the document's data and other information in the municipal data stream. This dependency creates an additional level of risk because it opens up the possibility that errors will be propagated through the system instead of isolated to a single document's record.

By outlining these automation tasks, we were able to identify a spectrum of implementation options that ranged from solving only a single task listed above, to a fully automated process with little or no human interaction. To evaluate the potential economic impact of different implementation strategies, we next considered a series of tasks where we began by solving the first task (2-class categorization) and then incrementally added each additional automation feature listed above. In practice, such a strict linear ordering is not necessary.

The results are displayed in Figure 1 as the "Reliance on Offshore Labor". *Reliance*, here, is calculated as the anticipated net change to manual encoding costs. Thus, the current approach indicated by the first bar, labeled "Manual" is normalized to 100%. Each subsequent bar shows the impact of an incremental automation step on the predicted cost of the remaining entries that must be made manually in the database. To calculate this cost, we need to know the number of fields required per document (20, for a fully manual approach), the number of fields that will be filled in automatically given a particular implementation, and the typical fraction of relevant documents amongst all the documents retrieved from a municipality (roughly 30% based on a sample of over 5000 documents.

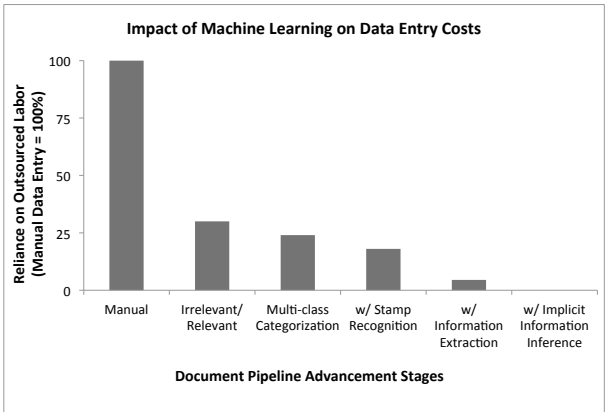

Figure 1: The potential impact of machine learning improvements on the document processing pipeline and the resulting decrease in reliance on outsourced labor.

Figure 1 clearly illustrates one critical point: solving the simplest problem (2-class categorization) has a very large economic impact. Importantly, this also happens to be the path of least commitment: a CEO anxious about relying heavily on machine learning might still be convinced to attempt solving a simple problem with an automated method, especially if such a solution could have a proportionally large impact on the business' bottom line.

Given the examination of the problem, we set off to see how much of the document processing task we could automate with off-the-shelf open-source technology. The overarching idea was to keep development costs extremely low with the knowledge that we would likely be able to solve at least the simple problem and make a significant economic impact.

We started from scratch with a small group of Master of Computer Science students and gave ourselves a one year timeline to tackle the problems. In the remaining sections of this paper, we describe some of the technologies we explored, the results, and the insights we obtained about the role of artificial intelligence and machine learning within this business domain.

Towards an Automated Document Pipeline

Initially, we constructed a corpus of 4441 public records documents. This corpus contained manually entered document type information for each record and pixel level markup for a subset of the records. Each document was processed using the open source Tesseract OCR platform (Smith 2007) to generate a text document from the scanned images. With this initial data set, we were able to explore multiple lines of inquiry: from the quality of avail-

able open source OCR software to our ability to find and recognize county recording stamps and our ability to classify documents appropriately.

We then explored each of the automation tasks described previously placing the most emphasis on those that were technologically straightforward and offered the highest economic yield.

2-Class Classification

We used the open-source Weka machine learning software (Hall et al. 2009) for the majority of our learning tasks. For document classification including both the two-class and multi-class problems, learning begins by transforming each multipage text document into a feature vector where features indicate the number of times a particular word occurs in the document [1]. We examined a range of classifiers using dictionaries from 200 to 800 words. Table 1 illustrates the accuracies achieved during ten-fold cross validation on our training set. All classifiers achieve accuracies above 97% and the best performer (DMNBtext (Su et al. 2008)) achieves nearly 99% accuracy with a 600 word dictionary. Note that with the exception of AdaBoost M1 using a 200 word dictionary, no classifier achieves a statistical victory over DMNBtext on any given dictionary size. Moreover, across all dictionary sizes, no classifier achieves a statistical victory over DMNBtext with a 600 word dictionary.

DMNBtext is a Discriminative Multinomial Naive Bayes classifier. Like all naive bayes classifiers, classification is based on the following application of Bayes' rule where c_j represents class label j and d_i represents the i^{th} document in the corpus $\mathcal{D}$ (McCallum, Nigam, and others 1998):

$$P(c_j|d_i) = \frac{P(c_j)P(d_i|c_j)}{P(d_i)} \tag{1}$$

Multinomial variants model $P(d_i|c_j)$ with the multinomial distribution using the following relationship in which word counts play a crucial role:

$$P(d_i|c_j) = P(|d_i|)|d_i|! \prod_{k=1}^{|\mathcal{V}|} \frac{P(w_k|c_j)^{n_{ik}}}{n_{ik}!} \tag{2}$$

In the expression above w_k is a word selected from a fixed vocabulary $\mathcal{V}$; and n_{ik} is the number of times w_k appears in d_i.

Since we are typically only interested in the most likely class, equations 1 and 2 can be combined and simplified so that one computes:

$$\arg\max_j P(c_j) \prod_{k=1}^{|\mathcal{V}|} P(w_k|c_j)^{n_{ik}} \tag{3}$$

The Bayes-optimal estimate for $P(w_k|c_j)$ used in equations 2 and 3 can be found simply by counting the number of times w_k occurs in the documents that comprise c_j in the training set (typically with Laplace smoothing to deal with edge conditions). Mathematically:

[1] For some classifiers, we use boolean feature vectors instead of a bag-of-words model.

$$P(w_k|c_j) = \frac{1 + \sum_{i=1}^{|D|} n_{ik} P(c_j|d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} n_{is} P(c_j|d_i)} \tag{4}$$

If a multinomial model is trained incrementally, one can maintain terms corresponding to the sums in the numerator and denominator of equation 4. When a new document from class c_j is obtained, the corresponding sums are updated based on the words observed in that document.

DMNBtext modifies this update/estimation procedure so that $P(w_k|c_j)$ is updated only *as needed*. Specifically, DMNBtext begins by estimating the sums in the numerator and denominator of equation 4 as 1 and $|V|$ respectively (the Laplacian estimator). Next, given a document d_i from the training set, DMNBtext *predicts* the most likely class using equation 1 and then computes the prediction loss as:

$$L(d_i) = P(c_j|d_i) - \hat{P}(c_j|d_i) \tag{5}$$

Where $\hat{P}(c_j|d_i)$ is the predicted probability that d_i is of type c_j and $P(c_j|d_i)$ is the actual probability. That is, $P(c_j|d_i)$ is 1 if and only if d_i is of type c_j; otherwise it is 0.

Then, instead of simply updating the sums in the numerator and denominator of equation 4 based on the number of times each word is observed, the update is weighted by the loss $L(d_i)$. Thus, if the classifier already makes a perfect prediction, no change to the term estimates will be made. On the other hand, the more the classifier's prediction differs from the ground truth, the greater the update and therefore the larger the contribution that d_i will have on modifying future predictions (since in this case, updates from d_i will have a large weight).

In a sense, DMNBtext combines the discriminative approach of the perceptron update rule with the statistical grounding of multinomial naive bayes. The modified parameter estimation method is a boon in environments like text classification where the independence assumption of Naive bayes is violated. While the standard multinomial approach tends to overweight correlated features, DMNBtext's discriminative approach is less susceptible to this problem and can achieve performance on par with SVM classifiers, but with a much faster, incremental, learning algorithm.

Multi-class Classification

Next, we explored multi-class classification after initial relevancy testing. We used an array of class-specific binary classifiers to identify the top three most frequently occurring relevant document subtypes: *Deeds of Trust*, *Statutory Warranty Deeds*, and *Quit Claim Deeds*. Although our number of training examples for each subtype was rather small (hundreds), we were able to obtain a very low false positive rate. This indicates that classifiers we obtained were appropriately specific. We were less concerned about false negative rate since the working assumption was that this would simply mean more documents that had to be manually classified.

Dict size (words)	200	400	600	800
DMNBtext (Su et al. 2008)	97.73	98.39	98.77	98.64
AdaBoost (J48) (Freund and Schapire 1996; Quinlan 1993)	98.29	98.46	98.56	98.59
RandomForest (Breiman 2001)	98.08	98.06	98.12	97.99
SPegasos (Shalev-Shwartz, Singer, and Srebro 2007)	97.88	97.87	97.87	97.67

Table 1: Accuracy of Top Classifiers

Document Type	Accuracy	F.N. Rate	F.P. Rate
Deed of Trust	96.80	0.041	0.026
Statutory Warranty Deed	98.72	0.035	0.008
Quit Claim Deed	98.80	0.062	0.006

Table 2: Document Sub-typing After Relevancy Testing

Stamp Classification

Figure 2 shows two documents from our training set. Both images have been stamped by the county office at the time of recording (these stamps are circled in the images). Recording stamps are significant because: (1) all counties have one; (2) many stamps contain information such as the recording date that must be extracted from the document to generate a database entry; and (3) many stamps contain a document type code that could improve document classification. A close-up view of a recording stamp from Chelan county is illustrated in Figure 3. This stamp has a number of valuable pieces of information including: the recording number, the date, and a document type code (DT in Figure 3). In addition, as we noted earlier, re-recorded documents can sometimes be identified because they carry two distinct recording stamps. Thus, by carefully probing for the recording stamp we hypothesized that we may be able to both improve document classification and extract some critical information.

To perform stamp classification, we first need to recognize the stamp. For this, a number of potential approaches are possible. While the bar code associated with most of the images present an obvious landmark, at the time of testing, the open source barcode detectors ZBar (Brown 2013) and zxing (Owen 2013) were not capable of correctly identifying the barcodes that appeared on all of the county recording stamps. We suspected this was either because of noise introduced during scanning or due to proprietary or obscure barcode formats. We then considered image template matching (Schweitzer, Deng, and Anderson 2011) and principal component analysis (Draper, Yambor, and Beveridge 2002) to find recording stamps directly from the pixel level data. However, we were able to obtain 100% accuracy using a simpler approach based on primitive features (e.g., height, width and density) of connected components.

Once the stamps were isolated, we used landmark features, defined for each county's stamp, to determine the placement of key features such as date, recording number and document type. These segments of text were then isolated and subjected to Tesseract's OCR.

Although the county specific document type data seems as though it will immediately solve the classification prob-

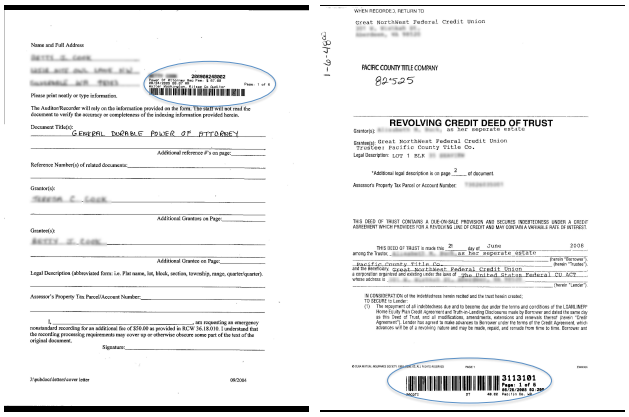

Figure 2: Two Documents from the Training Set (some information blurred to protect privacy)

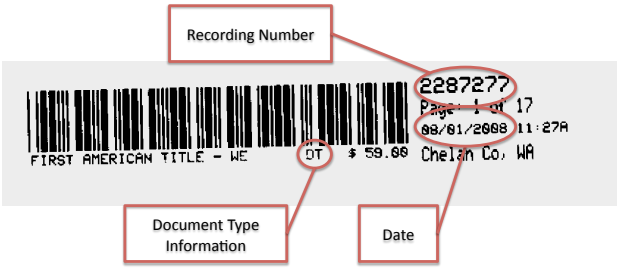

Figure 3: Close Up View of County Recording Stamp

Classifier / Modification	Accuracy	F.N. Rate
DMNB Text (600 words)	98.77%	1.86%
DMNB w/ whitelist	99.07%	1.44%
DMNB w/ whitelist and stamp rules	99.22%	1.12%

Table 3: Improved Classification

lem, this is not the case. As it happens, the document type code *and* the taxonomy of possible document type codes are *county-specific* and do not necessarily align with the document type taxonomy required by Data Data. Thus, we need to examine the county-specific taxonomies by extracting type codes using OCR (this process is imperfect especially given the small font typically used in these stamps) and then look for high support, perfect accuracy rules that predict document relevancy within our own taxonomical classification.

Not surprisingly, we cannot do this successfully in all situations or for all document sub-classes. Some county document codes are overly general and include both relevant and irrelevant documents. Other documents codes are easily misread by the OCR creating the false impression that a code is overly general. Despite all of these issues, it turns out we can create a set of targeted rules that make a statistically significant improvement to the 2-class classification performance in a subset of cases. Because of their limited applicability, they are not very useful by themselves. However, when combined with our previous DMNBtext classifier, they make a notable improvement in accuracy.

Table 3 shows the improved 2-class performance when a we augmented the 600 word dictionary with: (1) a whitelist of n-grams directly pulled from relevant document titles (e.g., "Deed of Trust" "Quit Claim"); and (2) a preprocessing step that uses high-support, perfect accuracy rules to initially screen documents based on their county-specific type code.

Information Extraction

While the recording stamp contained some important fields that would later be incorporated into the database entry, the typical font sizes used proved to be a limiting factor regarding the accuracy of off-the-shelf OCR. A number of possible approaches could be explored to improve accuracy including: (1) retraining the OCR software on the county-specific fonts used in the recording stamps; (2) performing image template matching (Schweitzer, Deng, and Anderson 2011) using a fixed set of county-specific templates instead of the more general OCR approach; or (3) looking for the desired data in other areas of the text document (i.e., outside of the recording stamp).

The first two options were relatively labor intensive as they meant training at the county level instead of the corpus level. Given this fact, and the fact that the bulk of the information we hoped to be able to extract was located in the document proper, we opted to explore the third option.

To understand the playing field, we used existing database entries for a number of documents in our corpus to directly search for matching terms within the OCR'd document. Since information extraction clearly relies on the documents containing the desired data to begin with, we wanted to verify that the data we were seeking was, in fact, inside each document. If we were unable consistently find the database entry within the text document, that would be an indication that either: (1) the data was obtained from some external source; or more likely that (2) the OCR software did not correctly interpret the image and thus produced text that prevented a match.

We chose seven fields from the database related to property addresses (e.g., Primary Address Number, City Name, etc). We selected these as we expected that all real-estate related documents would clearly indicate the property address and may even do so in more than one place, thereby offering some redundancy. Once we established an upper bound on performance, we trained a linear-chain Conditional Random Field (Lafferty, McCallum, and Pereira 2001) on a manually labeled set of documents to recognize the seven selected address fields. We then examined how well we were able to extract the appropriate labeled fields from within each document.

Table 4 illustrates the results of the information retrieval tasks. The first column indicates the database field under consideration. The second column indicates the fraction of documents from the test set from which the text from the database actually occurs within the OCR'd document. This represents an upper bound on performance. The third column represents the fraction of documents in which we were able to extract the same data from the OCR'd document as was represented in the database entry. This is recall. The final column indicates our recall performance relative to the upper bound.

The key findings from this table are three fold. First, the target data (aside from the State) is not consistently available from a direct search of the OCR'd text. In fact, the data from most critical fields: Primary Address Number and Street Name, can only be found in 92% and 82% of the test documents respectively. By itself, this is an indication that the performance of the OCR system in its off-the-shelf state is not up to the task of information extraction – at least not when considered under the standards of practice ($> 99\%$ accuracy). Of course, even these values are an upper bound on performance and our initial extractors built using Conditional Random Fields fall far short of identifying *all* of the address fields within a given document. So, it is not surprising that without a more extensive training corpus, our retrieval performance is well below the upper bound. Finally, when we consider these results along with the economic impact offered by information extraction, it seems clear that this task is best left to human labor—at least until we refine the OCR results and construct a significantly larger labeled training corpus.

Lessons and Insights

The case study explored in this paper presents some interesting lessons. Principle among these is that although data mining, machine learning and artificial intelligence are becoming household names, there are still substantial opportunities to introduce these technologies into the many business

Feature	Fraction of Data in Text	Fraction of Data Extracted	Performance Relative to Upper Bound
Primary Address Number	92%	55%	59%
Directional	71%	62%	88%
Street Name	82%	64%	78%
Suffix	97%	32%	33%
City Name	90%	74%	82%
State	100%	72%	72%
Zip Code	74%	71%	96%

Table 4: Performance of Conditional Random Field on Address Extraction

processes. Moreover, we believe that many of these business problems (such as the 2-class categorization) can: 1) be solved using well-established methods—in our case naive bayes classifiers; 2) be introduced without ceding all control to a learning algorithm—in our case, by retaining human input on the data extraction step; and 3) still have a significant and positive impact on the financial bottom line—in our case by reducing document processing by roughly 70%.

References

Breiman, L. 2001. Random forests. *Machine Learning* 45(1):5–32.

Brown, J. 2013. Zbar bar code reader. Available online: http://zbar.sourceforge.net/.

Draper, B. A.; Yambor, W. S.; and Beveridge, J. R. 2002. Analyzing pca-based face recognition algorithms: Eigenvector selection and distance measures. *Empirical Evaluation Methods in Computer Vision*.

Freund, Y., and Schapire, R. E. 1996. Experiments with a new boosting algorithm. In *Thirteenth International Conference on Machine Learning*, 148–156. San Francisco: Morgan Kaufmann.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I. H. 2009. The weka data mining software: An update. *SIGKDD Explor. Newsl.* 11(1):10–18.

Lafferty, J.; McCallum, A.; and Pereira, F. C. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*.

McCallum, A.; Nigam, K.; et al. 1998. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, 41–48. Citeseer.

Owen, Sean, e. a. 2013. Zxing. Available online: http://code.google.com/p/zxing/.

Quinlan, R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann Publishers.

Schweitzer, H.; Deng, R.; and Anderson, R. 2011. A dual-bound algorithm for very fast and exact template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33(3):459–470.

Shalev-Shwartz, S.; Singer, Y.; and Srebro, N. 2007. Pegasos: Primal estimated sub-gradient solver for svm. In *24th International Conference on MachineLearning*, 807–814.

Smith, R. 2007. An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition, 2007. ICDAR 2007.*, volume 2, 629–633.

Su, J.; Zhang, H.; Ling, C. X.; and Matwin, S. 2008. Discriminative parameter learning for bayesian networks. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, 1016–1023. New York, NY, USA: ACM.