Team Members:

Joseph Cappadona (PennKey: `jcapp`; Email: `jcapp@seas.upenn.edu`)

1 Introduction

1.1 The Problem

Information extraction is important to any data pipeline. But automating information extraction can be difficult, and there are many instances where manual data collection and entry is the only option. A robust document recognition and information extraction engine would vastly improve the rate at which certain types of data could be collected. Document classification and recognition can be used to extract data from forms, such as invoices [1], memos, and advertisements; from Heads-Up Diplays (HUDs), such as those used in mission simulators [2] and on sports broadcasts; and from web pages [3]. Current methods employ the use of layout analysis [4], context-free grammar learning [5], deep neural networks [3, 6, 7, 8], and classifiers like Naive Bayes, logistic regression, and linear SVM [1].

1.2 Data Sets

I plan to obtain data from the following sources:

1. Scanned Business Documents
   (a) Ghega-dataset, a dataset for document understanding and classification - `http://machinelearning.inginf.units.it/data-and-tools/ghega-dataset`
2. Heads-Up Displays
   (a) Ace Combat 5 (for PlayStation 2) Gameplay Footage - `https://www.youtube.com/watch?v=lDU-pccy6UM`
   (b) MLB The Show 16 (for PlayStation 4) Gameplay Footage - `https://www.youtube.com/watch?v=LoCjqdE5cdk`
3. Web Pages
   (a) Wikipedia Articles - `https://www.wikipedia.org`
   (b) Amazon Product Pages - `https://www.amazon.com`

The Ghega data set comes with document features already labeled, however data from the other sources will have to be labeled by hand.

2 Related Work

In [2], the authors built an image processing application with pattern recognition to analyze the HUD of Flight Test Campaigns and extract information important to the analysis of the campaign, such as aircraft position parameters, aircraft configuration information, tracking mode, and time.

Major limitations included its naive text recognition system, and image processing efficiency and text recognition accuracy, which were both largely due to the poor quality of the extracted HUD images. Additionally, this type of analysis is only effective for static, structured documents.

[5] approached information extraction from un-structured documents by utilizing automated context-free grammar learning and alignment-based learning. This type of grammar learning provides a dynamic, adaptive approach to information extraction that is necessary for schema learning. Similarly, [4] implemented layout analysis in order to learn common layout schemas for unstructured documents.

In [1], the authors built an unstructured document recognition system for business invoice processing centered around a bag-of-words approach that attempts to capture common layout and text features. They experimented with Naive Bayes, Multiclass (One-vs-All) Logistic Regression, and Multiclass (One-vs-All) Linear SVM. They were able to achieve 8.81% training error and 13.99% test error (through SVM) over a data set of 97 raw invoice images obtained from the internal testing library of the Oracle Corporation. They found F_1 score to be an effective means at comparing different models and found that l_2 regularization outperforms l_1 regularization for both logistic regression and linear SVM.

Comprehensive surveys such as [7, 8] demonstrate the potential for the use of deep neural networks for information extraction and document classification. However, these approaches rely on large repositories of documents, which can be problematic for certain use cases.

3 Problem Formulation

3.1 Problem Formulation

Given a set of documents X with associated sets of features Y, our goal is to train an algorithm h to, given a new document instance x, pull out textual features of interest $\widehat{f_i}$ and classify them into category $\widehat{c_i}$ to form a document feature $\widehat{y_i} := (\widehat{f_i}, \widehat{c_i})$. We will attempt to minimize the expected loss L, where

$$L(y, \widehat{y}) = \frac{1}{|F(x)|} \sum_{f_i \in F(x)} \mathbf{1}(h(\widehat{f_i}) \neq c_i)$$

or, in words, a loss of $\frac{1}{|F(x)|}$ is incurred for each textual feature $\widehat{f_i}$ that is misclassified, and a loss of 1 is incurred if all textual features for a given document are misclassified. We will assume that OCR will be consistent such that there is a one-to-one correspondence between features labeled during training and potential features identified during testing, and it is only the classification of the features that will be tested.

3.2 Data Processing

Data from the Ghega data set for document understanding and classification comes pre-annotated. Data from the Web Pages category will be accumulated by screenshotting select Wikipedia articles and Amazon product pages, running OCR on them, and hand-annotating key textual features. For data for the HUDs category, I will manually select stills from the videos in an attempt to simplify classification, run OCR on them, and hand-annotate key textual features.

This data will then be used to train the several different information extraction models as described below, and evaluate them according to the metrics described in the following section.

3.3 Performance Measures

Of course, for each machine learning method experimented with, I will report its training error and (cross-)validation error.

Additionally, I will seek to avoid overfitting my models by utilizing $L1$ and $L2$ regularization and evaluating the change in model performance with variation in these parameters.

Also, I plan to evaluate my models by computing F_1 scores. However, as described by [1], since the classification of different categories will likely behave very differently (e.g., dates and phone numbers will presumably be easier to extract than alphanumeric ID tags), instead of computing a single F_1 score for the entire model, we will compute the F_1 score for each classification category via a One-vs-All approach and then compute the average F_1 score across all categories, as these values provide a more wholistic metric for the model.

4 Solution Methods

[1] have experimented with Naive Bayes, Logistic Regression, and Linear SVM for unstructured document entity recognition. Despite Logistic Regression and Linear SVM clearly outperforming Naive Bayes, I plan to continue experimenting with all three methods to see if different methods work better in different scenarios with different feature generation procedures.

The work described above by [5, 4] on grammar- and layout-based learning will allow us to learn common structures among differently structured documents with similar content.

As described in [3, 6, 7, 8], Convolutional Neural Networks (CNNs) provide an effective means of learning document layouts and data schemas. I plan to experiment with these as well, however, other methods will be focused on due to the small size of the data sets I am working with, as well as due to the preference of learning human-understandable patterns (as opposed to the minimally human-understandable information that can be extracted from CNNs).

Additionally, I will experiment with k-Nearest Neighbor methods. For structured data from known documents, 1NN should be able to effectively detect which layout or information schema the document matches with. For documents with unstructured data and incomplete schemas, kNN on a global (considering the entire document) and local (considering only nearby entities) level could give us a good estimation of the layouts or schemas that best match the document being analyzed.

If time permits, I would also like to experiment with a boosting approach as detailed in [9]. Boosting permits us to use weak learners based on learned layout and schema constraints to boost classification performance.

5 Plan of Work

Rough outline of steps:

1. Implement simple bag-of-words feature extraction and classification using Naive Bayes, Multiclass Logistic Regression, and Multiclass SVM as described in [1]; test on new data sets; experiment with different feature generation procedures
2. Implement automated grammar/schema learning as described in [4] to learn document layouts and information schemas
3. Implement kNN to enhance schema matching
4. If time permits, experiment with CNNs to learn document layouts and information schemas
5. If time permits, implement a boosting approach to feature classification as described in [9] in an attempt to extract information from consistent, structured documents (like HUDs and Web Pages)

References

[1] Wenshun Liu, Billy Wan, and Yaqi Zhang. Unstructured document recognition on business invoice. *Stanford CS 229 Machine Learning Final Projects, Autumn 2016*, 2016.

[2] Luiz Eduardo Guarino de Vasconcelos, André Yoshimi Kusomoto, and Nelson Paiva Oliveira Leite. Using image processing and pattern recognition in images from head-up display. In *International Telemetering Conference Proceedings*. International Foundation for Telemetering, 2013.

[3] Tomas Gogar, Ondrej Hubacek, and Jan Sedivy. Deep neural networks for web page information extraction. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 154–163. Springer, 2016.

[4] Herve Dejean. Extracting structured data from unstructured document with incomplete resources. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 271–275. IEEE, 2015.

[5] Ramesh Thakur, Suresh Jain, Narendra S Chaudhari, and Rahul Singhai. Information extraction from the un-structured document using grammatical inference and alignment similarity. *Procedia Technology*, 4:365–369, 2012.

[6] Philipp Meerkamp and Zhengyi Zhou. Information extraction with character-level neural networks and noisy supervision. *arXiv preprint arXiv:1612.04118*, 2016.

[7] Shantanu Kumar. A survey of deep learning methods for relation extraction. *arXiv preprint arXiv:1705.03645*, 2017.

[8] Thien Huu Nguyen. *Deep Learning for Information Extraction*. PhD thesis, New York University, May 2017.

[9] Zachary Ives. Data and schema matching, April 2013.