


Cryptogram Decoding for Optical Character Recognition

Scott Leishman, University of Toronto
CIAR Summer School - August, 2006

Optical Character Recognition

- Classic pattern recognition problem with several commercial systems claiming better than 99% accuracy, *provided*:
 - machine printed text
 - noiseless, unskewed pages
 - “typical” script and language (for some systems)
 - “standard” font and point size used.
- Recognition quality quickly degrades when any of the above conditions fail to hold
- Systems typically use a labelled template matching approach with a large collection of models in various scripts, fonts, styles, and sizes
- How can we improve this?

Cryptogram Approach

- Determine connected blobs of ink in an image, then cluster similar blobs together
- Compare strings of cluster assignments with an underlying language lexicon based on cluster frequency and n-gram statistics to come up with an initial mapping from clusters to characters:
 - DGF LIHTOUFHW MPNNOHSG WI SCOUF AFTOHFMFHW EK0BF
GIKROHS P VRJNWISAPMQ
- Use confident mappings to guide refinement (like solving a cryptogram)
- Major advantage: Completely **font neutral**.
- Should work with any consistent “language” you can get statistics for (Java, Klingon, )

Our OCR Procedure

- Determine connected components using a two-pass algorithm [Haralick, Shapiro '92]:
 - Scan the pixels, propagating and assigning preliminary labels from neighbouring pixels, which are recorded in an equivalence table
 - Resolve the equivalence classes using DFS in the label equivalence graph
 - Scan and relabel based on the resolved equivalence classes

0	1	1	0	0	0	1
0	0	1	1	1	1	1
1	0	0	0	0	0	1
1	1	0	0	0	1	0
1	0	0	0	0	1	0
0	1	0	1	0	1	0
0	0	1	1	1	1	0

input image



0	A	A	0	0	0	B
0	0	A	A	A	A	A
C	0	0	0	0	0	A
C	C	0	0	0	A	0
C	0	0	0	0	A	0
0	C	0	D	0	A	0
0	0	C	C	C	A	0

labels after first sweep



	A	B	C	D
A	1	1	1	0
B	1	1	0	0
C	1	0	1	0
D	0	0	1	1

equivalence table



0	A	A	0	0	0	A
0	0	A	A	A	A	A
A	0	0	0	0	0	A
A	A	0	0	0	0	A
A	0	0	0	0	0	A
0	A	0	A	0	A	0
0	0	A	A	A	A	0

labels after 2nd sweep

we can implicitly integrate out the infinitely many

Our OCR Procedure

- Throw out any components with large aspect ratio or small size (horizontal lines, small punctuation)
- Process components page by page, initially adding each to its own cluster, then refine by a straight Euclidean distance match
- Refine remaining clusters iteratively by attempting to split and merge cluster averages, and by carrying out either a convolved Euclidean or Hausdorff distance match



Hausdorff Distance Matching

- Ideally want to group **e, e, e, e** all under the same cluster, Euclidean distance doesn't take into account how far mismatched pixels are from one another

- Hausdorff “distance” from image X to Y

$$h(X, Y) = \max_{x \in X} \min_{y \in Y} d(x, y)$$

where $d(x, y)$ is Euclidean or another distance metric

2.2	1.4		0	0	0	0	0	0	0		1.4
1.4		0	0	0			0	0	0	0	
	0	0			1.4	1.4		0	0	0	
	0	0						0	0	0	
	0	0	0	0	0	0	0	0	0	0	0
	0	0									
0	0	0		2	2	2	2	2	2	2	2
	0	0		1.4	2.2	2.8	2.8	2.2	1.4		
	0	0	0		1.4	2	2	1.4		0	0
	0	0	0	0					0	0	
1.4		0	0	0	0	0	0	0	0		1.4
2.2	1.4		0	0	0	0	0	0		1.4	2.2

2.2	1.4		0	0	0	0	0	0	0		1.4
1.4		0	0	0			0	0	0	0	
	0	0			1.4	1.4		0	0	0	
	0	0						0	0	0	
	0	0	0	0	0	0	0	0	0	0	0
	0	0									
0	0	0		2	2	2	2	2	2	2	2
	0	0		1.4	2.2	2.8	2.8	2.2	1.4		
	0	0	0		1.4	2	2	1.4		0	0
	0	0	0	0					0	0	
1.4		0	0	0	0	0	0	0	0		1.4
2.2	1.4		0	0	0	0	0	0		1.4	2.2

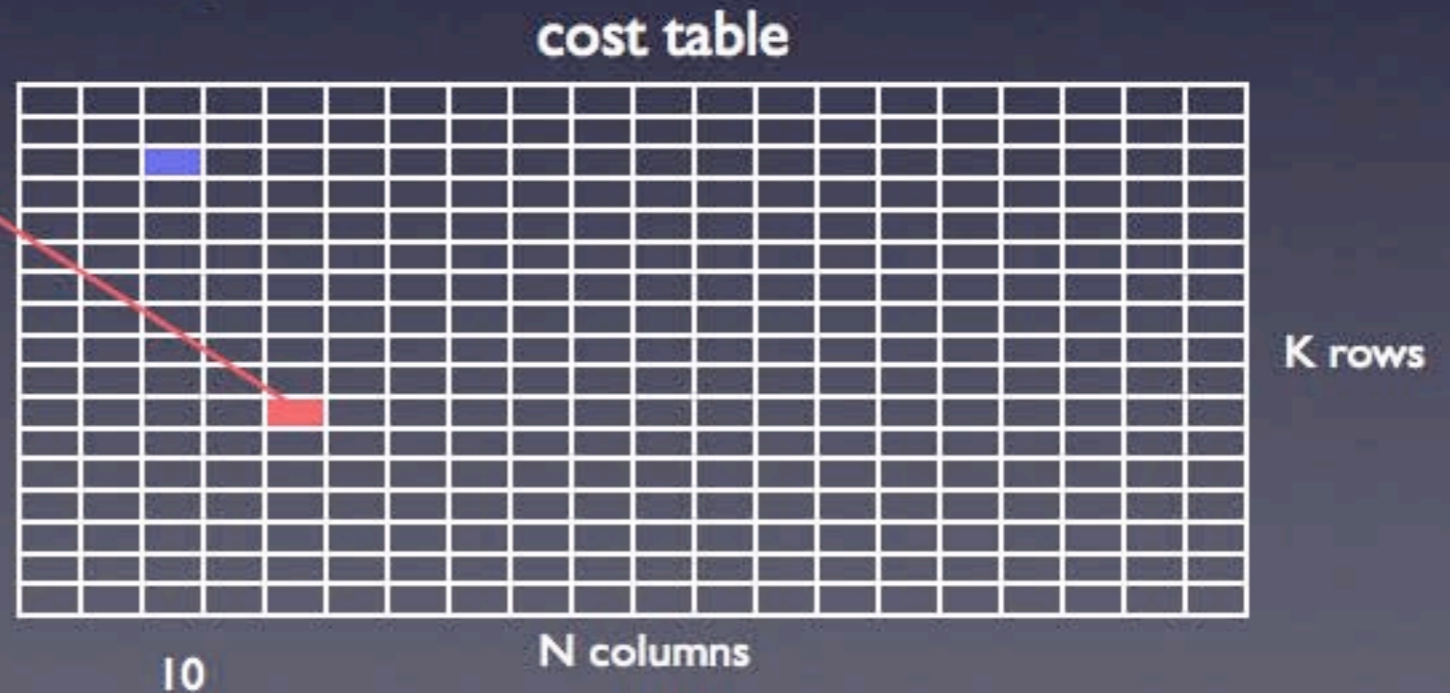
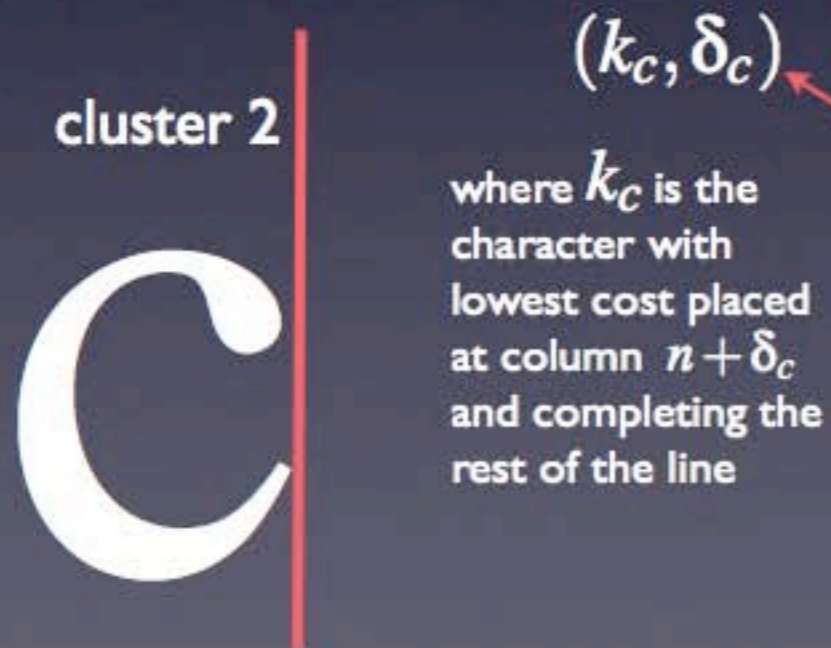
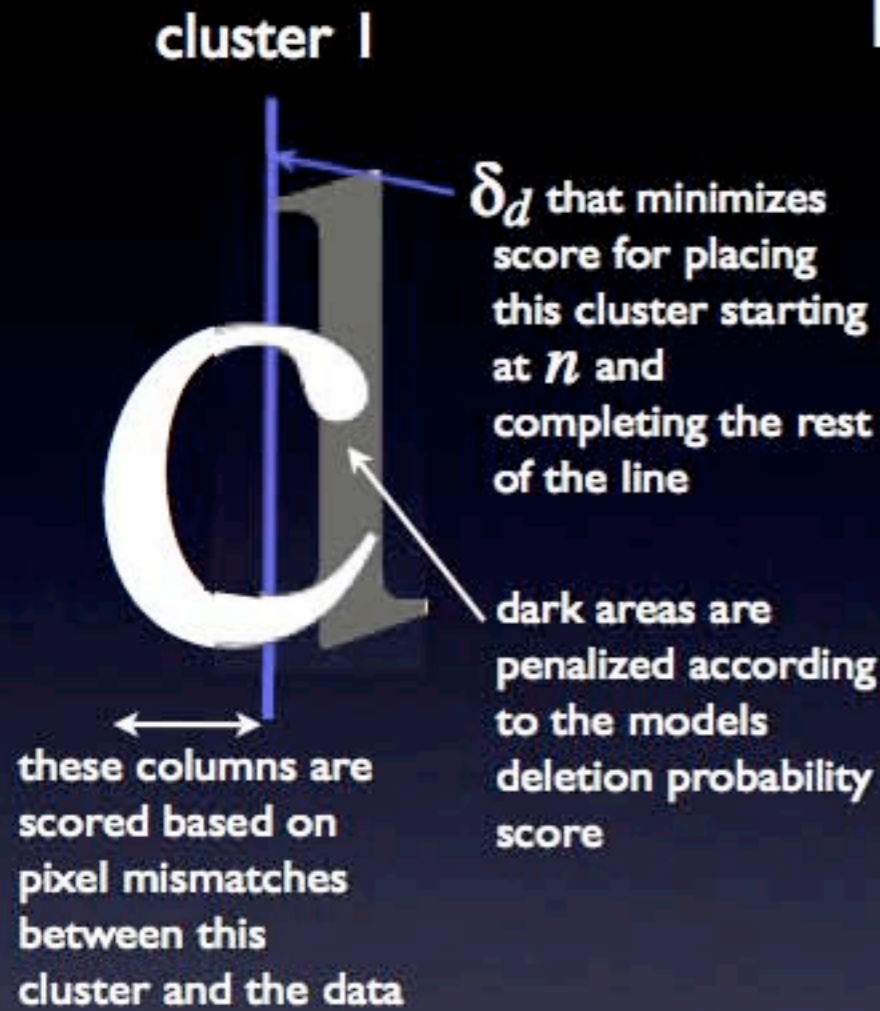
Line Solving

- Dynamic Programming used to fill in entries of a cost table. Columns of the table correspond to the columns of the image, and rows of the table correspond to characters
- Each table entry defines the cost of placing a character c down starting at column n

$$cost(c, n) = \min_{k, \delta} [(\text{model score of } c \text{ from } n \text{ to } n + \delta) + \text{bigram cost from } c \text{ to } k + cost(k, n + \delta + 1)]$$

- Model score is based on mismatches between the model pixels and the underlying data pixels
- Limited to **character bigrams** only

Line Solving



Character Decoding Issues

- Cryptogram decoding assumes there is a one-to-one mapping between clusters and characters, this isn't always the case
- Ligatures Ex: 'fi' result in a single connected component. Characters like 'i', 'j' are made up of 2 components
- Can augment the output alphabet to include an upper bound on the number of components in a character: $\Sigma' = \{\varepsilon\} \cup \Sigma \cup \Sigma^2$

αost ghost
friαten → frighten

!337\$p34k

- Huang et al (unpublished) Have performed synthetic tests using the Leetspeak “language”
- Able to achieve 86% character and 64% word accuracy (but 99% on ascii coded text)

a	@, 4	h	}{, #	o	0	v	v
b	8, b	i	!, i	p	9, o	w	w
c	c, [j	j	q	q, 0_	x	><, x
d	d	k	k	r	r, 2	y	y
e	3, e	l	1, l	s	\$, z	z	%
f	ph	m	(V), m	t	7, t		
g	g, 6	n	n	u	u, v		

gold is expected to continue its rise this year due to renewed inflationary pressures especially in the us

g01d !\$ ex|oect3d t0 [0n7!nve i7z ri\$e 7# !\$ y3@2 due t0 r3new3d !nphl@t!0n@2y |orezzur3z ez9eci4lly in t#e uz

(Near) Future Work

- **Still lots to be done! Very much a work in progress.**
- **Accuracy improvements during clustering (better merges, splits and matches)**
- **Speed improvements in line solving (beam search) and Hausdorff matching**

The Bigger Picture

- **JTAG system for finding and classifying regions of document images into 25 different categories [Laven et al '05]**
- **Region information currently used as a preprocessor to determine text regions upon which to run our OCR pipeline**
- **Ideally would like to use OCR results to improve region identification, and use region class to modify n-grams and character distributions (ex. mathematical expressions versus code blocks versus plain text)**

abstract

bullet_item

decoration

eq_number

figure

figure_label

footnote

header

main_title

references

subsection_heading

table_caption

text

ion nodes in a manner to greedily minimize

$$\sum_{\text{leaf } j} \left(p_j^+ \ln \left(\frac{p_j^- + p_j^+}{p_j^+} \right) + p_j^- \ln \left(\frac{p_j^-}{p_j^- + p_j^+} \right) \right)$$

are the fraction of positive and negative examples at leaf j is then $\text{sign}(p_j^+ - p_j^-)$. Viewed differently, we assign a real number f_j at each leaf with the intention of minimizing the empirical loss associated with logistic regression

$$\sum_{\text{leaf } j} \left(p_j^+ \ln(1 + e^{-2f_j}) + p_j^- \ln(1 + e^{2f_j}) \right)$$

and, over choices of f_j , when $f_j = (1/2) \ln(p_j^+ / p_j^-)$ and thresholding f_j gives the hard prediction rule. This can be viewed as a margin-based learning algorithm. The algorithm is logistic regression

software_list

code_block

editor_list

equation

figure_caption

footer

graph

image

pg_number

section_heading

table

table_label

EM-DD: An Improved Multiple-Instance Learning Technique

Qi Zhang

Department of Computer Science
Washington University
St. Louis, MO 63130-4899
qz@cs.wustl.edu

Sally A. Goldman

Department of Computer Science
Washington University
St. Louis, MO 63130-4899
sg@cs.wustl.edu

Abstract

We present a new multiple-instance (MI) learning technique (EM-DD) that combines EM with the diverse density (DD) algorithm. EM-DD is a general-purpose MI algorithm that can be applied with boolean or real-value labels and makes real-value predictions. On the boolean Musk benchmarks, the EM-DD algorithm without any tuning significantly outperforms all previous algorithms. EM-DD is relatively insensitive to the number of relevant attributes in the data set and scales up well to large bag sizes. Furthermore, EM-DD provides a new framework for MI learning, in which the MI problem is converted to a single-instance setting by using EM to estimate the instance responsible for the label of the bag.

1 Introduction

The *multiple-instance* (MI) learning model has received much attention. In this model, each training example is a set (or *bag*) of instances along with a single label equal to the maximum label among all instances in the bag. The individual instances within the bag are not given labels. The goal is to learn to accurately predict the label of previously unseen bags. Standard supervised learning can be viewed as a special case of MI learning where each bag holds a single instance. The MI learning model was originally motivated by the *drug activity prediction problem* where each instance is a possible conformation (or shape) of a molecule and each bag contains all likely low-energy conformations for the molecule. A molecule is active if it binds strongly to the target protein in at least one of its conformations and is inactive if no conformation binds to the protein. The problem is to predict the label (active or inactive) of molecules based on their conformations.

The MI learning model was first formalized by Dietterich et al. in their seminal paper [4] in which they developed MI algorithms for learning axis-parallel rectangles (APRs) and they also provided two benchmark "Musk" data sets. Following this work, there has been a significant amount of research directed towards the development of MI algorithms using different learning models [2,5,6,9,12]. Maron and

EM-DD: An Improved Multiple-Instance Learning Technique

Qi Zhang

Department of Computer Science
Washington University
St. Louis, MO 63130-4899
qz@cs.wustl.edu

Sally A. Goldman

Department of Computer Science
Washington University
St. Louis, MO 63130-4899
sg@cs.wustl.edu

Abstract

We present a new multiple-instance (MI) learning technique (EM-DD) that combines EM with the diverse density (DD) algorithm. EM-DD is a general-purpose MI algorithm that can be applied with boolean or real-value labels and makes real-value predictions. On the boolean Musk benchmarks, the EM-DD algorithm without any tuning significantly outperforms all previous algorithms. EM-DD is relatively insensitive to the number of relevant attributes in the data set and scales up well to large bag sizes. Furthermore, EM-DD provides a new framework for MI learning, in which the MI problem is converted to a single-instance setting by using EM to estimate the instance responsible for the label of the bag.

1 Introduction

The *multiple-instance* (MI) learning model has received much attention. In this model, each training example is a set (or *bag*) of instances along with a single label equal to the maximum label among all instances in the bag. The individual instances within the bag are not given labels. The goal is to learn to accurately predict the label of previously unseen bags. Standard supervised learning can be viewed as a special case of MI learning where each bag holds a single instance. The MI learning model was originally motivated by the *drug activity prediction problem* where each instance is a possible conformation (or shape) of a molecule and each bag contains all likely low-energy conformations for the molecule. A molecule is active if it binds strongly to the target protein in at least one of its conformations and is inactive if no conformation binds to the protein. The problem is to predict the label (active or inactive) of molecules based on their conformations. The MI learning model was first formalized by Dietterich et al. in their seminal paper [4] in which they developed MI algorithms for learning axis-parallel rectangles (APRs) and they also provided two benchmark "Musk" data sets. Following this work, there has been a significant amount of research directed towards the development of MI algorithms using different learning models [2,5,6,9,12]. Maron and

Raton [7] applied the multiple-instance model to the task of recognizing a person from a series of images that are labeled positive if they contain the person and negative otherwise. The same technique was used to learn descriptions of natural scene images (such as a waterfall) and to retrieve similar images from a large image database using the learned concept [7]. More recently, Ruffo [11] has used this model for data mining applications. While the musk data sets have

boolean labels, algorithms that can handle real value labels are often desirable in real-world applications. For example, the binding affinity between a molecule and receptor is quantitative, and hence a real-value classification of binding strength is preferable to a binary one. Most prior research on MI learning is restricted to concept learning (i.e. boolean labels). Recently, MI learning with real-value labels has been performed using extensions of the diverse density (DD) and k -NN algorithms [1] and using MI regression [10]. In this paper, we present a general-purpose MI learning technique (EM-DD) that combines EM [3] with the extended DD [1] algorithm. The algorithm is applied to both boolean and real-value labeled data and the results are compared with corresponding MI learning algorithms from previous work. In addition, the effects of the number of instances per bag and the number of relevant features on the performance of EM-DD algorithm are also evaluated using artificial data sets. A second contribution of this work is a new general framework for MI learning of converting the MI problem to a single-instance setting using EM. A very similar approach was also used by Ray and Page [10].

2 Background

Dietterich et al. [4], presented three algorithms for learning APRs in the MI model. Their best performing algorithm (*iterated-discrimin*), starts with a point in the feature space and "grows" a box with the goal of finding the smallest box that covers at least one instance from each positive bag and no instances from any negative bag. The resulting box was then expanded (via a statistical technique) to get better results. However, the test data from Musk1 was used to tune the parameters of the algorithm. These parameters are then used for Musk1 and Musk2. Auer [2] presented an algorithm, MULTINST, that learns using simple statistics to find the halfspaces defining the boundaries of the target APR and hence avoids some potentially hard computational problems that were required by the heuristics used in the iterated-discrimin algorithm. More recently, Wang and Zucker [11] proposed a lazy learning approach by applying two variants of the k nearest neighbor algorithm (k -NN) which they refer to as citation- k NN and Bayesian k -NN. Ramon and De Raedt [9] developed a MI neural network algorithm. Our work builds heavily upon the *Diverse Density* (DD) algorithm of Maron and Lozano-Pérez [5,6]. When describing the shape of a molecule by n features, one can view each conformation of the molecule as a point in a n -dimensional feature space. The diverse density at a point p in the feature space is a probabilistic measure of both how many *different* positive bags have an instance near p , and how far the negative instances are from p . Intuitively, the diversity density of a hypothesis h is just the likelihood (with respect to the data) that h is the target. A high diverse density indicates a good candidate for a "true" concept. We now formally define the general MI problem (with boolean or real-value la

els) and DD likelihood measurement originally defined in [6] and extended to real-value labels in [1]. Let D be the labeled data which consists of a set of m bags $B = \{B_1, \dots, B_m\}$ and labels $L = \{\ell_1, \dots, \ell_m\}$, i.e., $D = \{ \langle B_1, \ell_1 \rangle, \dots, \langle B_m, \ell_m \rangle \}$. Let bag $B_i = \{B_{i1}, \dots, B_{ij}, \dots, B_{in}\}$ where B_{ij} denote the j^{th} in $\{\ell_1, \dots, \ell_m\}$, i.e., D distance in bag i . Assume the labels of the instances in B_i are $\ell_{i1}, \dots, \ell_{ij}, \dots, \ell_{in}$. For boolean labels, $\ell_i = \max\{\ell_{i1}, \ell_{i2}, \dots, \ell_{in}\}$. The diverse density of hypothesized target point h is $DD(h) = \Pr\{h | \dots \vee \ell_{in}\}$, and for real-value labels, ℓ_i is defined as $DD(h) = \Pr\{h |$

Related Work

- Using cryptograms for OCR goes back at least to 1986 (Nagy)
- Zhang, Zhou, and Tygar (2005) had an interesting paper that used a \$10 microphone, clustering keyboard click sounds to recognize typed text. After a 12-20 minute recording session their system could recognize over 96% of typed characters, and correctly guess 90% of 5 character passwords in fewer than 20 attempts!