Old English Characters Recognition Using Neural Networks

Sattajit Sutradhar

Georgia Southern University
Department of Mathematical Sciences

Masters Thesis Defense
(under the direction of Ionut E. Iacob)
June 1st, 2018.

Outline

1. Character Recognition
2. Artificial Neural Networks
3. Classification
4. Character-image Recognition
   - Implementation
   - Experimental results
5. Conclusion
6. References

History and Applications

- Early character recognition can be traced back to 1914.
- In the late 1920s into 1930s a “Statistical Machine” has been developed: searching microfilm archives using an optical code recognition system
- Optical Character Recognition (OCR) is widespread nowadays:
  - Help visually impaired persons.
  - Automatic number plate recognition (Georgia Southern uses one!).
  - Automatic data entry.
  - Converting handwriting (on a computer touch screen) into typed text (hard!).
  - Make images of printed documents searchable.

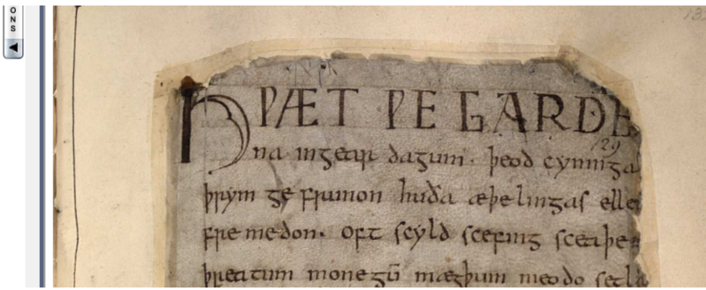
History and Applications

- Library of Congress launched an effort in 2005 to create the "World Digital Library" (with at least one private contribution of \$3 million)
  - Guess who got the money?!
- Huge benefits
  - Easy information access.
  - Rare and unique cultural materials made available to more than one billion people.
  - Help preserving rare materials.
  - Search support.

Handwritten manuscripts

- Not everything can be easily automated

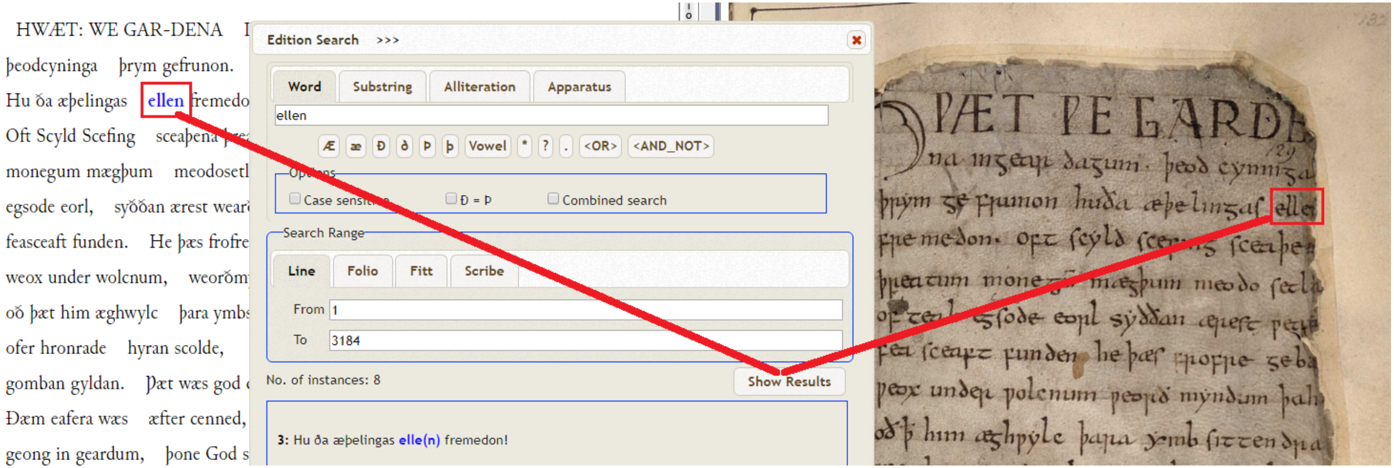
| HWÆT: WE GAR-DENA IN GEARDAGUM
þeodcyninga þrym gefrunon.
Hu ða æþelingas ellen fremedon!
Oft Scyld Scefing sceaþena þreatum
monegum mægþum meodosetla ofteah,
egsode eorl, syððan ærest wearð
feasceaft funden. He þæs frofre gebad,
weox under wolcnum, weorðmyndum þah,



- Information was/is manually extracted

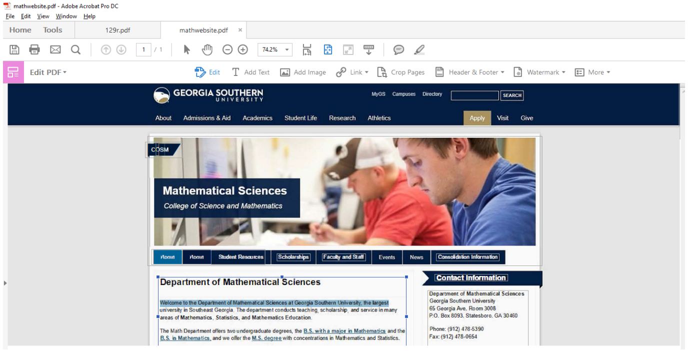
Handwritten manuscripts

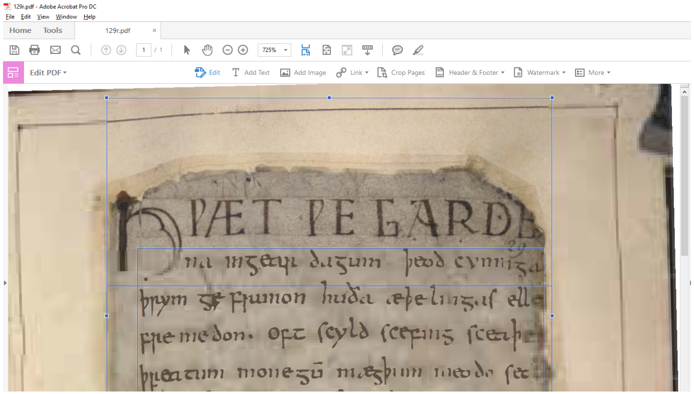
- Searching information makes sense in both text and image contexts



- Goal: easy, practical way to perform character recognition in manuscript images.

- Off-the-shelf tools: Adobe Acrobat can perform excellent OCR in images!





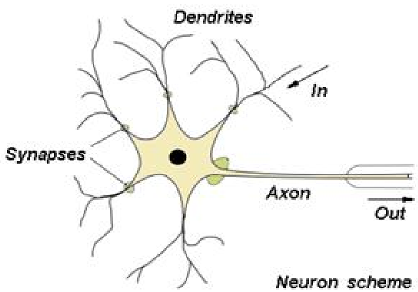
- Poor performance for character recognition in manuscript images.

Outline

1. Character Recognition
2. Artificial Neural Networks
3. Classification
4. Character-image Recognition
   - Implementation
   - Experimental results
5. Conclusion
6. References

Neural networks

- How does brain work?
- Dendrites Collect information.
- The information then passed through the synapses.
- Cell body processes the information.
- Axons receive the information and conveys to the dendrites of the next nerve cell.



Neural networks

- Early implementation of the idea:

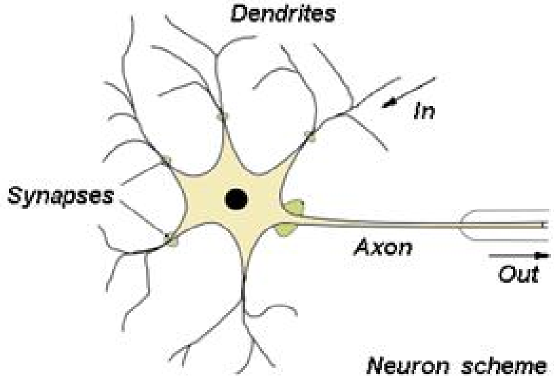


Figure 1: Human nerve cell

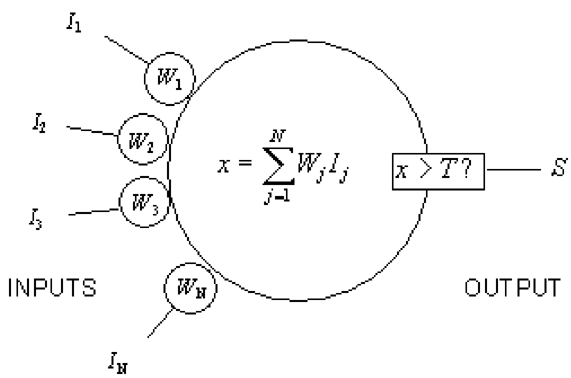
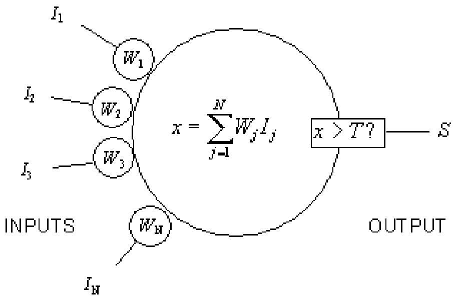


Figure 2: Neural Network

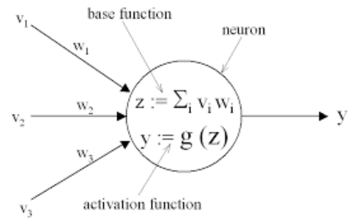
- Dendrites can be compared to input units, cell body to nodes, synapses to activation function and axons to the output units.
- In 1943, neurophysiologist Warren McCulloch and mathematician Walter Pitts wrote a paper on how neurons might work.

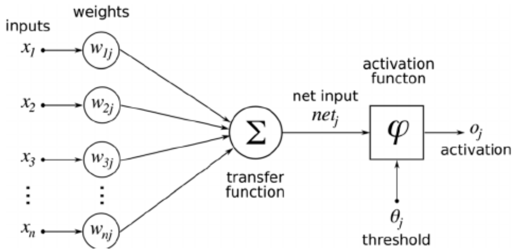
Artificial Neural Networks (ANN)

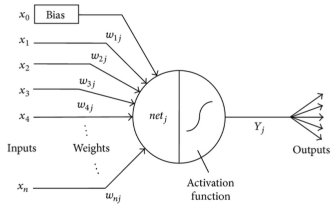
- McCulloch and Pitts modeled a simple neural network using electrical circuits.Neurons being the basic computational unit of the brain.
- The advance of computers in the 1950's made possible first simulations.
- Bernard Widrow and Marcian Hoff of Stanford developed ADALINE and MADALINE(Multiple ADAptive LINear Elements, 1959).
- In 1962, Widrow and Hoff developed a learning procedure, examining the value before the weight adjusts it.



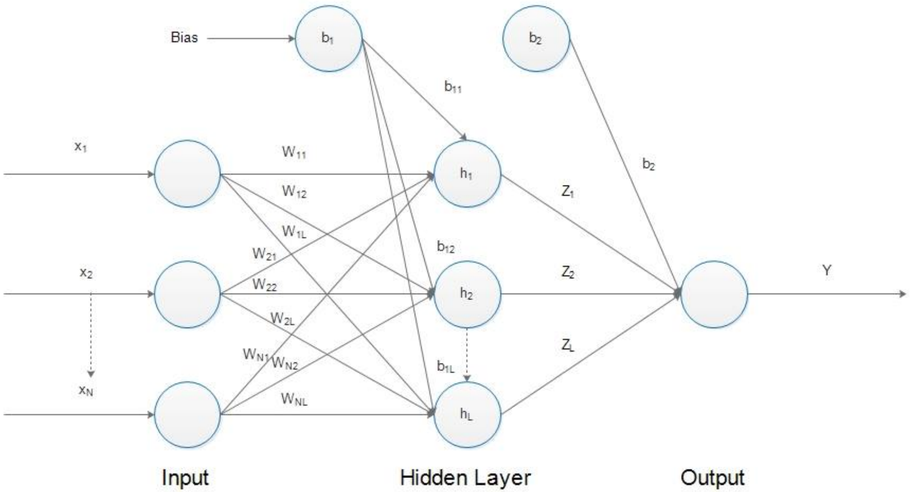
- The model captured moderate interest and was refined through the 80's.
- However, with the development of von Neumann model and research on Neural network was left behind even though John von Neumann himself suggested the imitation of neural functions.
- A breakthrough in 1986: backpropagation algorithm (D. E. Rumelhart, C. E. Hinton, and R. J. Williams)







Generic ANN, one hidden layer



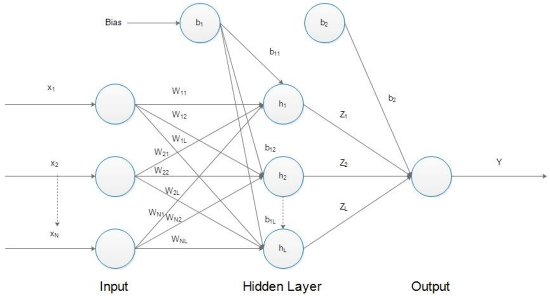
Generic ANN, one hidden layer

- Mathematical Model

$$y : \mathbb{R}^N \to (0,1)$$

$$y(\mathbf{x}; \mathbf{w}, \mathbf{b}) =$$

$$\sigma\left(\sum_{j=1}^{L} z_j \sigma\left(\sum_{i=1}^{N} w_{ij} x_i + b_{1j}\right) + b_2\right)$$



where $\sigma()$ is the sigmoid function:

$$\sigma : \mathbb{R} \to \mathbb{R}, \qquad \sigma(x) = \frac{1}{1+e^{-x}}$$

Outline

1. Character Recognition
2. Artificial Neural Networks
3. Classification
4. Character-image Recognition
   - Implementation
   - Experimental results
5. Conclusion
6. References

Classification

- In machine learning and statistics, classification is the supervised learning task in which the computer learns from the input data and then uses this learning to create a model and classify new observation.
- Supervised Machine Learning
  - Input variables x and output variables Y are known.
  - Goal: To use an algorithm to learn the mapping function from the input to the output, $Y = f(X)$ well enough.
  - Learning stops when the algorithm achieves an acceptable level of performance.
- A classification model generates conclusion based on the observed data set . Given inputs will attempt to predict the value of one or more labeled outcomes.

Classification

- Two Categories
  - Bi-class classification: classifying between two categories.
    - Examples: Pass or fail, sick or not sick etc.
  - Multi-class classification: classifying more than two categories.
    - Examples: Classifying a set of images, handwriting recognition etc.
- Various classification algorithms exist:
  Linear Classifiers (Logistic Regression, Naive Bayes Classifier); Support Vector Machines; Decision Trees; Boosted Trees; Random Forest; Neural Networks; K-Nearest Neighbor.

Definition (Classification in Euclidian Space)

Let $R, B \subseteq \mathbb{R}^n$ be finite sets. The binary classification problem consists in finding a function $f : \mathbb{R}^n \to \{-1, 1\}$ such that

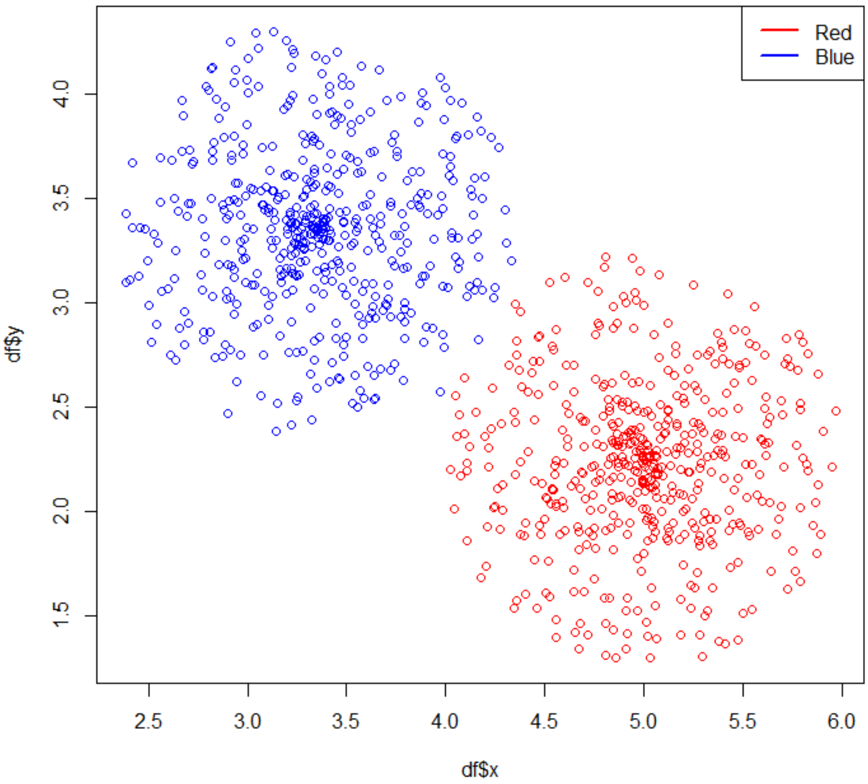
$$f(x) = \begin{cases} 1, & \text{if } x \in R \\ -1, & \text{if } x \in B \end{cases}$$

One possible solution:

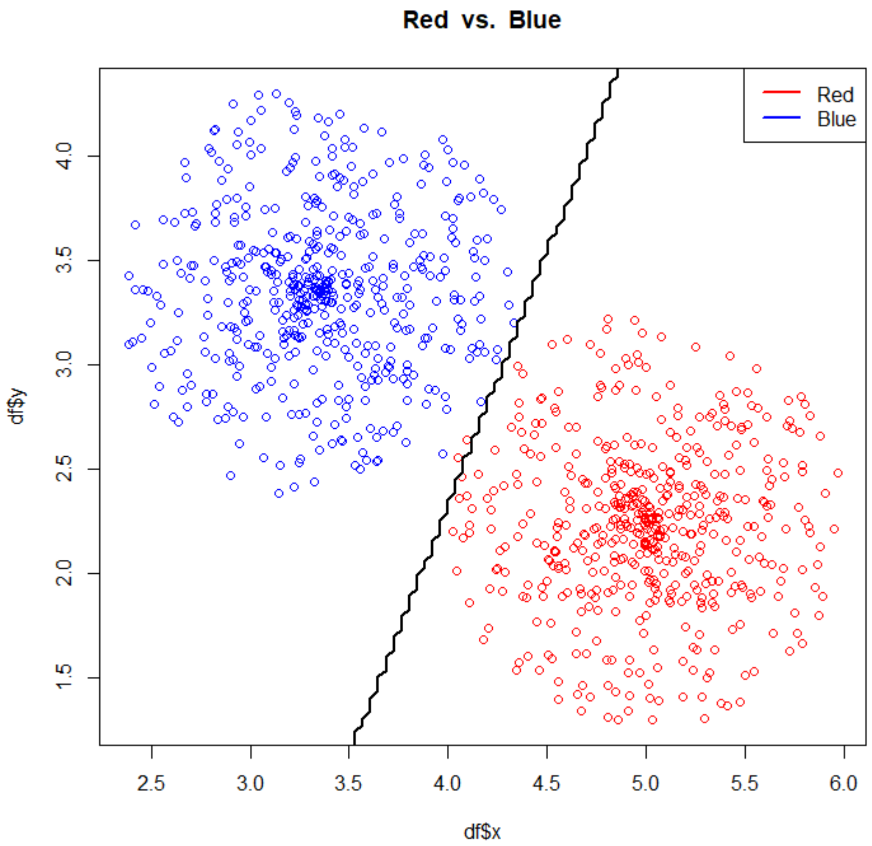
- Find a hyperplane $\boldsymbol{w}^T\boldsymbol{x} - w_0 = 0$, where $\boldsymbol{w} \in \mathbb{R}^n$ is a constant vector (n components), $w_0 \in \mathbb{R}$, and $\boldsymbol{x} \in \mathbb{R}^n$ is a variable (n components), such that:

$$\begin{aligned} \boldsymbol{w}^T\boldsymbol{s} - w_0 > 0, & \quad \forall \boldsymbol{s} \in R \\ \boldsymbol{w}^T\boldsymbol{t} - w_0 < 0, & \quad \forall \boldsymbol{t} \in B \end{aligned}$$

Classification Example



Classification Example



Multi-class classification

Definition (Multi-class Classification in Euclidian Space)

Let us consider C finite, disjoint sets $S_1, S_2, \ldots, S_C \subseteq \mathbb{R}^n$. The multi-class classification problem consists in finding a function $f_C : \mathbb{R}^n \to \{1, 2, \ldots, C\}$ such that

$$f_C(x) = \begin{cases} 1, & \text{if } x \in S_1 \\ 2, & \text{if } x \in S_2 \\ \ldots & \\ C, & \text{if } x \in S_C \end{cases}$$

Outline

1. Character Recognition
2. Artificial Neural Networks
3. Classification
4. Character-image Recognition
   - Implementation
   - Experimental results
5. Conclusion
6. References

- Convolutional Neural Networks: perform sophisticated image recognition/classification.
  - Fairly difficult to use and set up
  - May require significant amount of data for training
  - Very computationally intensive

Image Extraction

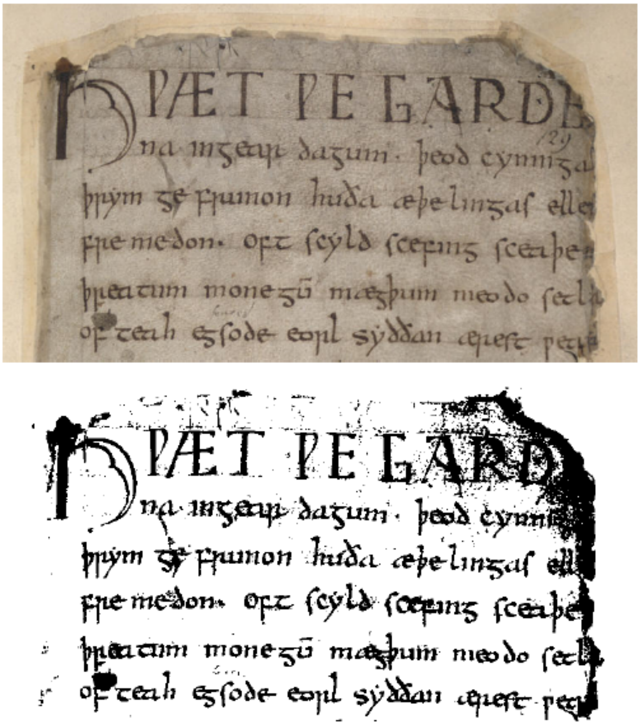


Figure : Manuscript image: original (top) and converted to B&W (bottom)

Image Extraction

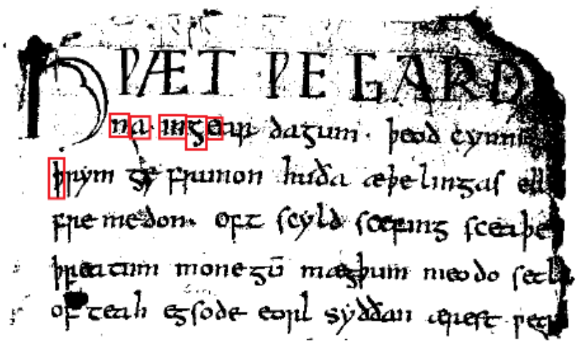


Figure : Extracting character images from a B&W manuscript image

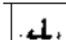
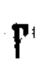
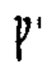
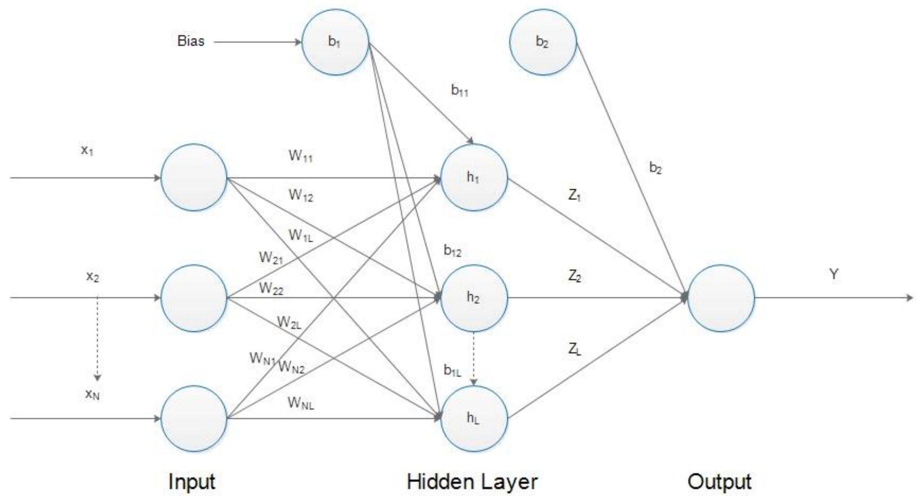
Char	Character Images			
a			...	
b			...	
...	...	...	...	...
w			...	

Table : The set $\mathcal{I}$ of all lower case character images in manuscript

Implementation

- NN takes input as a vector $\vec{x} \in \mathbb{R}^n$ whereas our input is an image.
- How to use an image (2D) as input for a NN (vector)?



$$\Rightarrow \begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

Algorithm 1 Image normalization algorithm

procedure MAT($img \in \mathcal{I}$) ▷ computes image matrix
Input: $img \in \mathcal{I}$
Output: $M \in \mathbb{R}_{p \times q}$
    $M \leftarrow img$ pixel values
    while first column of M only zeros **do**
        remove first column of M
    end while
    while first row of M only zeros **do**
        remove first row of M
    end while
    append rows with zeros up to p rows
    append columns with zeros up to q columns
    return M ▷ Returns the normalized image matrix
end procedure

 $\Rightarrow \begin{bmatrix} 1 & 0 & 1 & \ldots & 0 \\ 1 & 1 & 1 & \ldots & 0 \\ 1 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ldots & \vdots & \\ 0 & 0 & 0 & \ldots & 0 \end{bmatrix}$

-
- Matrix vectorization methods:
  - by rows
  - by columns
  - using singular value decomposition (SVD) and the (left/right) singular vectors
  - computing the (columns/rows) correlation matrix, then vectorized by rows

Definition

The character image classification model M is represented by the commutative diagram:

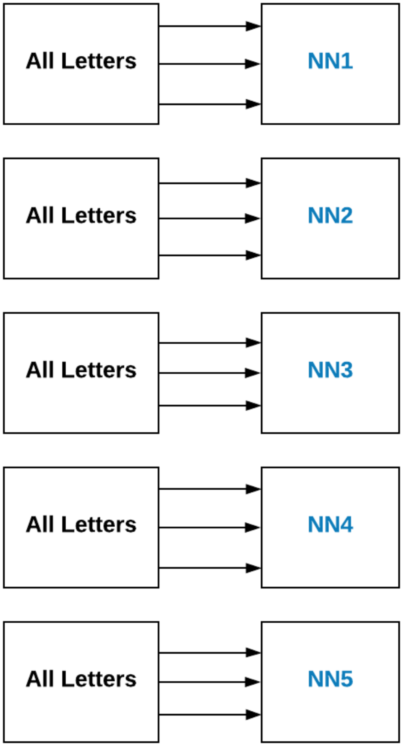
$$
\begin{tikzcd}
\mathcal{I} \arrow[r, "mat"] \arrow[drr, "M"'] & \mathbb{R}_{p\times q} \arrow[r, "vec"] & R^n \arrow[d, "\widehat{F_C}"] \\
& & \mathcal{L}
\end{tikzcd}
$$

Experimental results

- Implemented in R v3.3.3 using the *neuralnet* library, running on Windows 10, 64-bit Intel Core i7 CPU @3.40GHz, 16GB RAM.
- We used 20 samples of each of the 22 letters: "a", "ae", "b", "c", "d", "e", "eth", "f", "g", "h","i", "l", "m", "n", "o", "p", "r", "s", "t", "thorn", "u", "w".
- Experimented with groups of 4-5 letters from the list of all letters, with different number of hidden layers and neurons

Experimental Procedure and results

Training Of Neural Networks



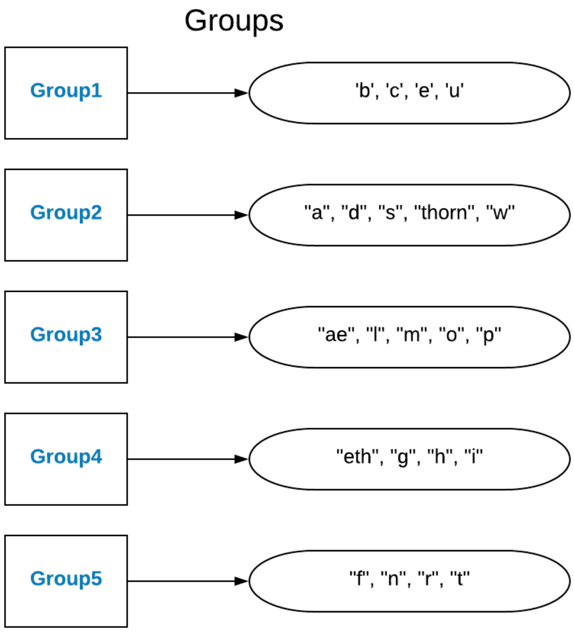
Selection of Groups

Original \ Predicted	a	ae	b	c	d	e	et	f	g	h	i	l	m	n	o	p	r	s	t	th	u	w
a	2	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ae	0	1	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
b	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
d	2	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
et	0	0	0	0	2	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
f	0	0	0	0	0	0	0	3	0	0	0	0	0	1	0	0	0	0	0	0	0	0
g	0	1	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0
h	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
i	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0
n	0	0	0	0	0	0	0	0	0	0	1	0	2	1	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0
r	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	1
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
t	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	1	0	0	0
th	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0
u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
w	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3

Algorithm 2 Incremental learning algorithm

procedure INCREMENTLEARN($\mathcal{I}$) ▷ computes models for groups of letters in $\mathcal{I}$
Input: $\mathcal{I}$
Output: $(N_i, \mathcal{L}_i)$
    $\mathcal{L} \leftarrow$ list of all letters
    $i \leftarrow 1$ ▷ Group counter
    while letters in $\mathcal{L}$ **do**
        create best model for all letters in $\mathcal{L}$
        $\mathcal{L}_i \leftarrow$ *subset of* $\mathcal{L}$ ▷ Select group of 4-6 best classified letters
        remove $\mathcal{L}_i$ from $\mathcal{L}$
        $N_i \leftarrow$ model for $\mathcal{L}_i$
        $i \leftarrow i + 1$ ▷ Increment group counter
    end while
    return $(N_i, \mathcal{L}_i)$ ▷ Returns all groups of letters and their models
end procedure

Experimental Procedure and results



```
[1] "Letters: b, c, e, u, zz"
[1] "Layers: 32"
[1] "Confusion Table"
          Predicted
Original  b  c  e  u zz
      b   2  0  0  0  2
      c   0  4  0  0  0
      e   0  0  1  0  3
      u   0  0  0  4  0
      zz  0  1  0  1 70
[1] "Accuracy: "
[1] 0.9204545455
```

Figure 3: Model predictions results for Group 1: b, c, e, u, zz (with $zz = other$)

```
[1] "Letters: a, d, s, thorn, w, zz"
[1] "Layers: 32"
[1] "Confusion Table"
        Predicted
Original  a  d  s thorn  w zz
   a      2  0  0     0  0  2
   d      1  2  0     0  0  1
   s      0  0  4     0  0  0
   thorn  0  0  0     4  0  0
   w      0  0  0     0  3  1
   zz     0  0  1     0  1 66
[1] "Accuracy: "
[1] 0.9204545455
```

Figure 4: Model predictions results for Group 2: $a, d, s, thorn, w, zz$ (with $zz = other$)

```
[1] "Letters: ae, l, m, o, p, zz"
[1] "Layers: 32"
[1] "Confusion Table"
         Predicted
Original ae  l  m  o  p zz
      ae  1  0  0  1  0  2
      l   0  4  0  0  0  0
      m   0  0  4  0  0  0
      o   0  0  1  2  0  1
      p   0  0  0  0  1  3
      zz  3  0  2  0  0 63
[1] "Accuracy: "
[1] 0.8522727273
```

Figure 5: Model predictions results for Group 3: ae, l, m, o, p, zz (with $zz = other$)

```
[1] "Letters: eth, g, h, i, zz"
[1] "Layers: 32, 10"
[1] "Confusion Table"
        Predicted
Original eth  g  h  i zz
     eth   3  0  0  0  1
     g     0  2  0  0  2
     h     0  0  2  0  2
     i     1  0  0  2  1
     zz    0  1  0  2 69
[1] "Accuracy: "
[1] 0.8863636364
```

Figure 6: Model predictions results for Group 4: eth, g, h, i, zz (with $zz = other$)

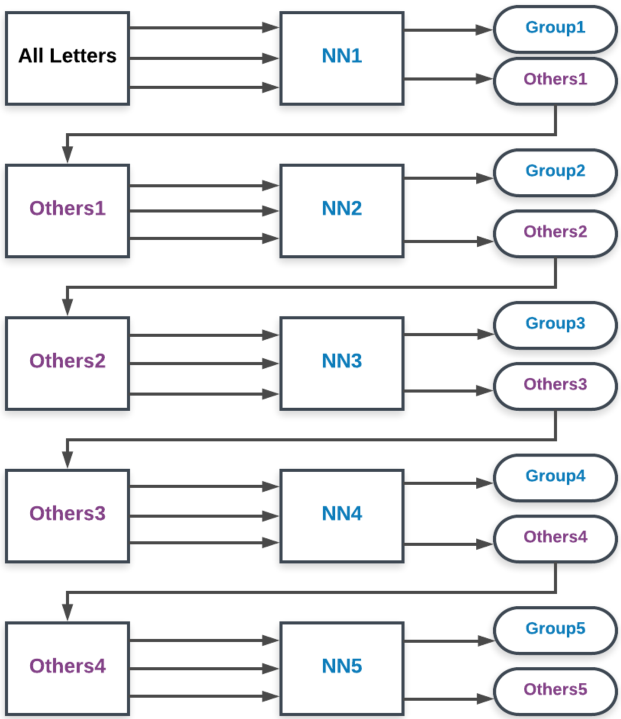
```
[1] "Letters: f, n, r, t, zz"
[1] "Layers: 64, 30, 12"
[1] "Confusion Table"
        Predicted
Original  f  n  r  t zz
      f   1  1  1  0  1
      n   0  2  0  0  2
      r   0  0  1  0  3
      t   0  0  0  2  2
      zz  0  0  1  0 71
[1] "Accuracy: "
[1] 0.875
```

Figure 7: Model predictions results for Group 5: f, n, r, t, zz (with $zz = other$)

Algorithm 3 Hierarchical multi-class classification algorithm

procedure HierarchicalMultiClass($img \in \mathcal{I}$) ▷ multi-class prediction
Input: $img \in \mathcal{I}, (N_i, \mathcal{L}_i)$ ▷ an image, all models and letter groups
Output: $letter$ ▷ $letter$ classification of img
    $letter \leftarrow NONE$
    for each $(N, \mathcal{L})$ in list $(N_i, \mathcal{L}_i)$ **do**
        $letter \leftarrow N(img)$ ▷ Classify img with model N
        if $letter \in \mathcal{L}$ **then**
            return $letter$ ▷ classification of img found
        end if
    end for
    $letter \leftarrow$ best classification among all N_i's
    return $letter$ ▷ Returns the classification of img
end procedure

Classifying Machine



Experiments

1. Experiment 1: we performed multi-class classification of all letter images using direct vectorization by columns of the image matrix.
2. Experiment 2: we performed and compared multi-class classification of all letter images using various vectorization techniques and choose SVD.
3. Experiment 3: we performed multi-class classification by groups of letters, using the best classification method identified in Experiment 2.

Experiment1 Result

Table 4.1: The confusion table for results in Experiment 1 (Accuracy: 0.6136363636)

		Predicted																				
		a	ae	b	c	d	e	et	f	g	h	i	l	m	n	o	p	r	s	t	th	u	w
Original	a	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0
	ae	0	2	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	b	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	c	0	0	0	3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
	d	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0
	e	0	0	0	0	0	3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	et	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	f	0	0	0	0	0	0	0	2	0	0	1	0	0	0	0	0	0	0	0	0	0	1
	g	0	0	0	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	h	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	1	0	1	0	0	0	0
	i	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0
	l	0	1	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0
	m	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0
	n	1	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0
	o	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0
	p	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	2
	r	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0
	s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
	t	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	0
	th	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0
	u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
	w	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

Experiment2 Result

Table 4.2: The confusion table for results in Experiment 2 (Accuracy: 0.6590909091)

Original \ Predicted	a	ae	b	c	d	e	et	f	g	h	i	l	m	n	o	p	r	s	t	th	u	w
a	2	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ae	0	1	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
b	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
d	2	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
et	0	0	0	0	2	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
f	0	0	0	0	0	0	0	3	0	0	0	0	0	1	0	0	0	0	0	0	0	0
g	0	1	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0
h	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
i	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0
n	0	0	0	0	0	0	0	0	0	0	1	0	2	1	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0
r	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	1
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
t	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	1	0	0	0
th	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0
u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
w	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3

Experiment3 Results

Table 4.3: The confusion table for results in Experiment 3 (Accuracy: 0.7613636364)

Original	Predicted																					
	a	ae	b	c	d	e	et	f	g	h	i	l	m	n	o	p	r	s	t	th	u	w
a	2	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ae	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
b	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
c	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
d	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
e	0	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eth	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
f	0	0	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0
g	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	1	0	0	0
h	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	1	0	0	0
i	0	0	0	0	0	0	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0
n	0	0	0	0	0	0	0	0	0	0	1	0	1	2	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	1	0	0	0
p	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	0	1	0	0	0
r	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	3	0	0	0	0	0
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
t	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	0	0	0
thorn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0
u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
w	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	3

Outline

1. Character Recognition
2. Artificial Neural Networks
3. Classification
4. Character-image Recognition
    - Implementation
    - Experimental results
5. Conclusion
6. References

Conclusion

- We are proposing a practical, accurate, and computationally efficient method for Old English characters recognition from manuscript images.
- Our method relies on a modern machine learning model, Artificial Neural Networks, to perform character recognition based on individual character images cropped directly from manuscript pages.
- We propose model dimensionality reduction methods that improve accuracy and computational effectiveness.
- Our experimental results show that the model we propose outperforms previous attempts as well as current automatic text recognition techniques.

Future Work

- Humanities scholars working with manuscripts typically perform an initial manual text extraction from manuscript images, followed by adding various metadata information from images or editorial work.
- This work can be extended two-fold: (i) Automatic text extraction from manuscript images and
  (ii) Combining edited manuscript textual information with character images recognition (described in this work) to produce more accurate character recognition for directly searching manuscript images.

References

McCP W. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity, Bulletin of Mathematical Biophysics, Vol. 5, 1943, 115-133.

Ros Rosenblatt, F. (1962). Principles of Neurodynamics, Spartan Books, New York.

RHW Rumelhart, D.E., Hinton, G.E., and Williams, R.J. (1986). Learning internal representations by backpropagating errors. Nature, 323:533–536.

EBeo Kevin Kiernan, Ionut E. Iacob, Electronic Beowulf, online edition (2018), http://ebeowulf.uky.edu/