Large Scale Character Classification

Neeba N.V

ADVISOR
PROF. C. V. JAWAHAR

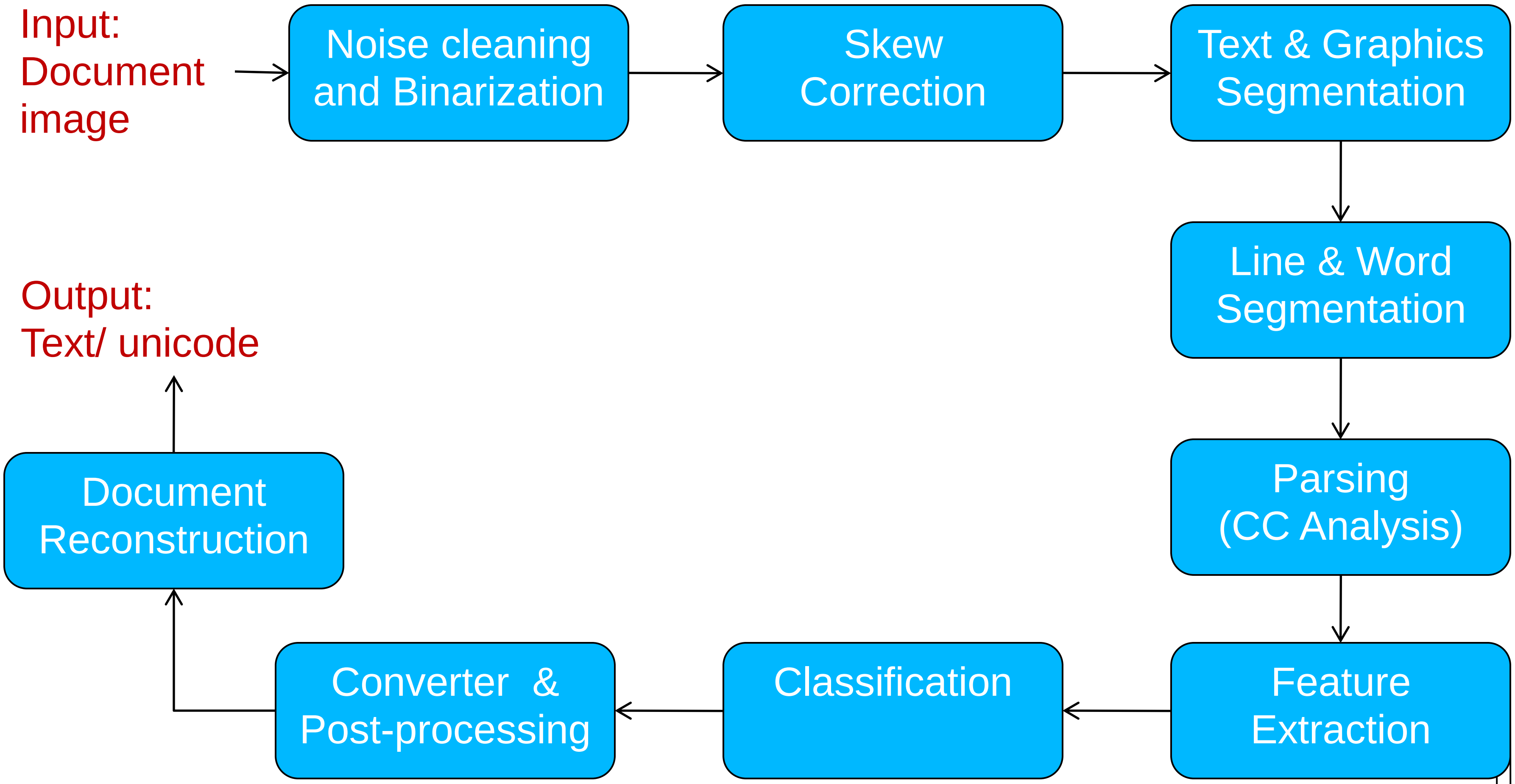
Pattern Classification

- Given a sample x.
  - Find the label corresponding to it.

- A classifier is an algorithm, which takes x and returns the label between 1 to N.
  - Binary Classification -- N = 2
  - Multiclass classification -- N > 2

- Evaluation is usually done as probability of correct classification.

Overall architecture of an OCR system



Focus of this Thesis

- Classification of characters/patterns for a large class (number of classes in the order of hundreds) problem.
- We choose character recognition for an Indic language, Malayalam, as our area.
- However, our methods are highly generic (language and script independent).
- Conducted experiments on a large real dataset.

Challenges for Indic OCR

- Large number of characters.
- Large number of similar/confusing characters.
- Complex character graphemes.
- Unicode/display/font related issues.
- Variation in glyph of a character with change in font/style.
- Lack of standard databases, statistical information and benchmarks for testing.
- Lack of well developed language models.
- Quality of documents in terms of paper quality, print quality, age of document, the resolution of scanning.
- Appearance of foreign or unknown symbols.

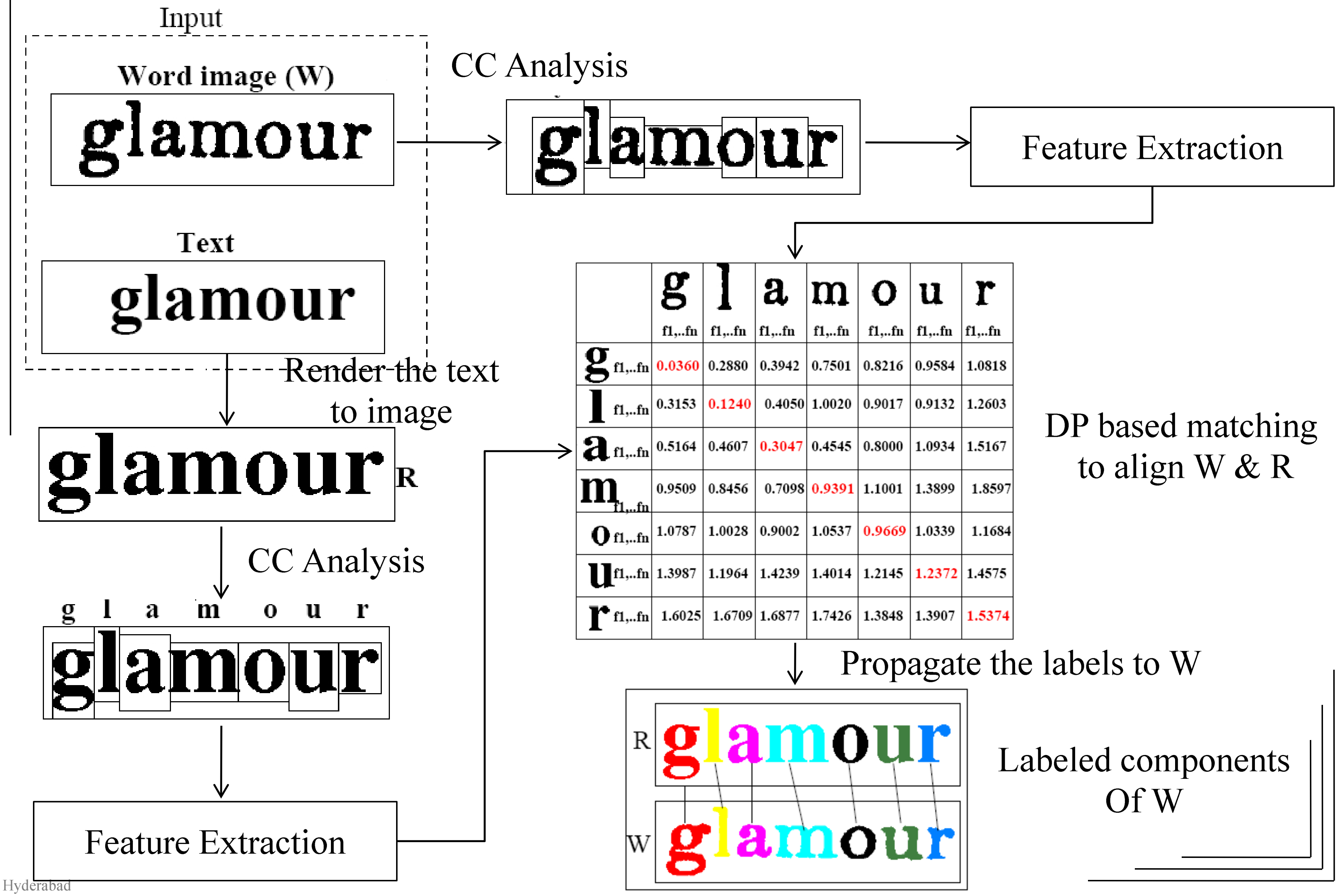
Challenges Specific to Malayalam Script

- Non-Standard Font Design.
- Script Variations.
- Representation Issues.
- Compound Words and Dictionaries.

Building Datasets from Real Life Documents

- Large dataset for training and testing OCR.
- Symbol level annotated data.
- Challenges
  - Degradation : Cuts, merges, spurious noise etc.
  - Language specific issues.
  - Font and Unicode related issues
- Our Solution :
  - Dynamic Programming based word alignment algorithm.

Alignment of an English Word



Input

Word image (W)

glamour

Text

glamour

Render the text to image

glamour R

CC Analysis

g l a m o u r

glamour

Feature Extraction

CC Analysis

glamour

Feature Extraction

	g f1,..fn	l f1,..fn	a f1,..fn	m f1,..fn	o f1,..fn	u f1,..fn	r f1,..fn
g f1,..fn	0.0360	0.2880	0.3942	0.7501	0.8216	0.9584	1.0818
l f1,..fn	0.3153	0.1240	0.4050	1.0020	0.9017	0.9132	1.2603
a f1,..fn	0.5164	0.4607	0.3047	0.4545	0.8000	1.0934	1.5167
m f1,..fn	0.9509	0.8456	0.7098	0.9391	1.1001	1.3899	1.8597
o f1,..fn	1.0787	1.0028	0.9002	1.0537	0.9669	1.0339	1.1684
u f1,..fn	1.3987	1.1964	1.4239	1.4014	1.2145	1.2372	1.4575
r f1,..fn	1.6025	1.6709	1.6877	1.7426	1.3848	1.3907	1.5374

DP based matching to align W & R

Propagate the labels to W

R glamour

W glamour

Labeled components Of W

Algorithm to align Indic Scripts

1. Input: Word image W and the corresponding Unicode/text from annotation.
2. Convert the Unicode to the class labels using a map file MAP.
3. Reorder the symbols, using the language rules in RULES file.
4. Render the symbols to get a word R, and label each symbol with the corresponding class label.
5. Find the connected components in the original word image.
6. Initialize the dynamic programming table D of size m × n, where (m − 1) and (n − 1) are the no. of connected components in R and W respectively.
7. Fill each cell, D(i, j) in the table using the following equation.

   $$D(i,j) = \min \begin{cases} D(i-1,j-1) + MC(R_i, W_j) \\ D(i-1,j) + MC((R_{i-1}, R_i), W_j) \\ D(i,j-1) + MC(R_i, (W_{j-1}, W_j)) \end{cases}$$

   where, MC(Ri,Wj) is the matching Cost of symbol Ri in the text(rendered as image) with symbol Wj in the original image.
8. Get the matching String by reconstructing the path, by following the minimum cost path.
9. Propagate the labels of symbols in R to W.

Alignment of a Malayalam Word

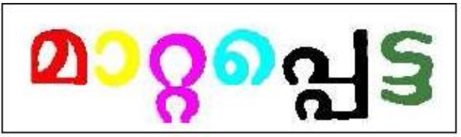
Input

Word Image(W)

Unicode

U+0d2e;U+0d3e;U+0d31;U+0d4d;
U+0d31;U+0d2a;U+Od4d;U+Od2a;
U+Od46;U+0d1f;U+0d4d;U+0d1f;

CC Analysis



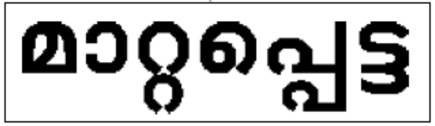
Feature Extraction

Convert to symbols
Reorder symbols

33; 51; 122; 52; 113; 107

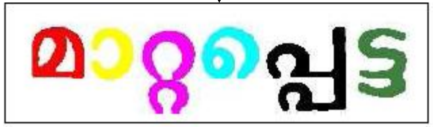
MAP FILE
RULES FILE

Render the word image



R

CC Analysis
Labels the CCs

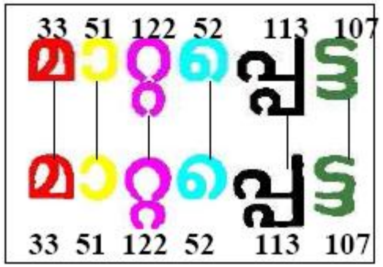


33 51 122 52 113 107

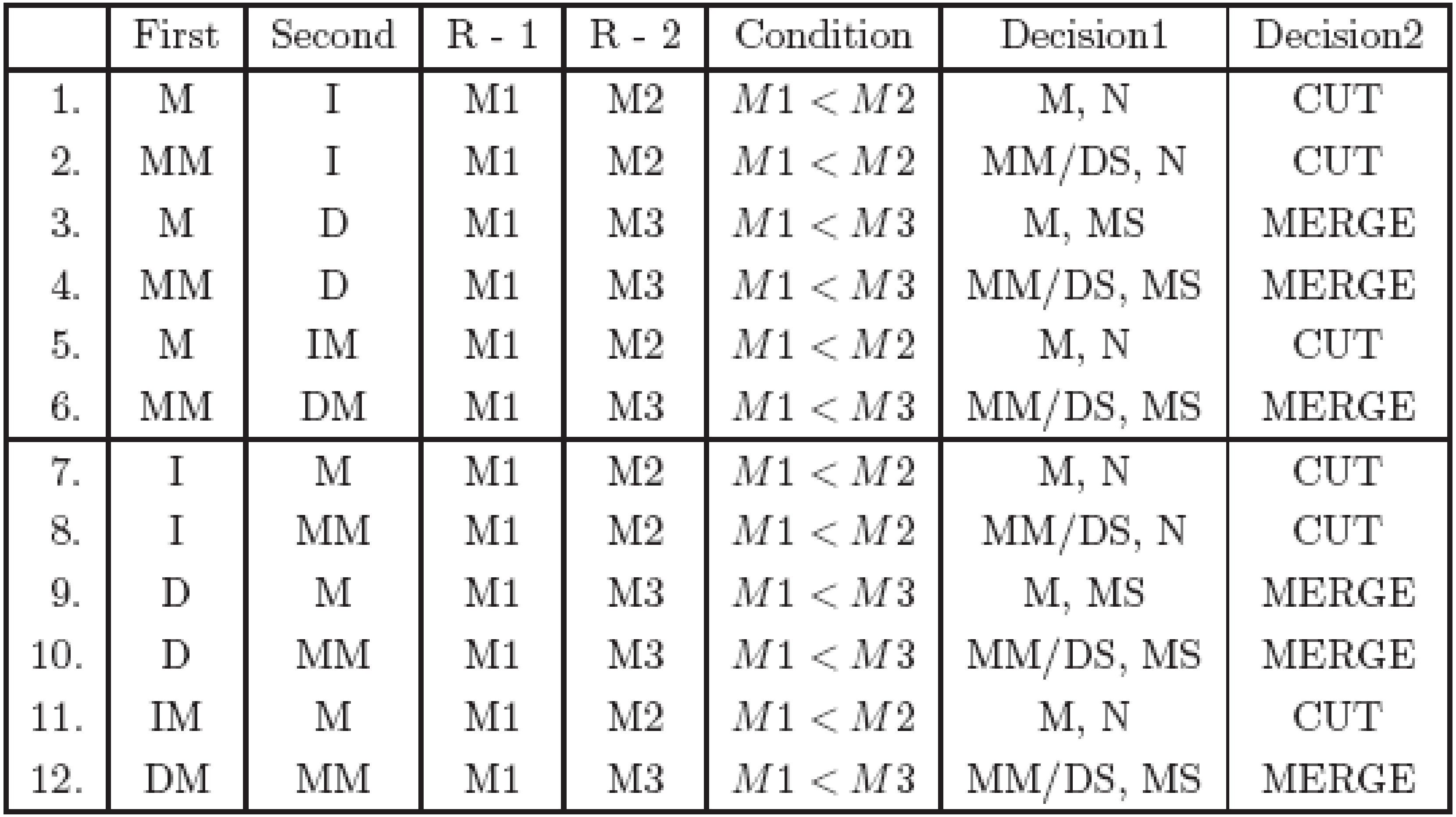
Feature Extraction

	മ f1..fn	ാ f1..fn	റ്റ f1..fn	െ f1..fn	പ്പ f1..fn	ട്ട f1..fn
മ f1..fn	0.2627	0.8975	1.5988	2.1796	3.0155	3.8445
ാ f1..fn	0.6491	0.3015	0.3789	0.4750	0.8455	0.9330
റ്റ f1..fn	1.2455	0.3923	0.3297	0.3956	0.7832	0.8481
െ f1..fn	1.6744	0.5550	0.4316	0.3940	0.8123	0.9251
പ്പ f1..fn	2.1506	0.7079	0.5976	0.5754	0.6156	0.7751
ട്ട f1..fn	2.6993	0.8335	0.6790	0.7102	0.9488	0.6396

DP based Matching to align R and W

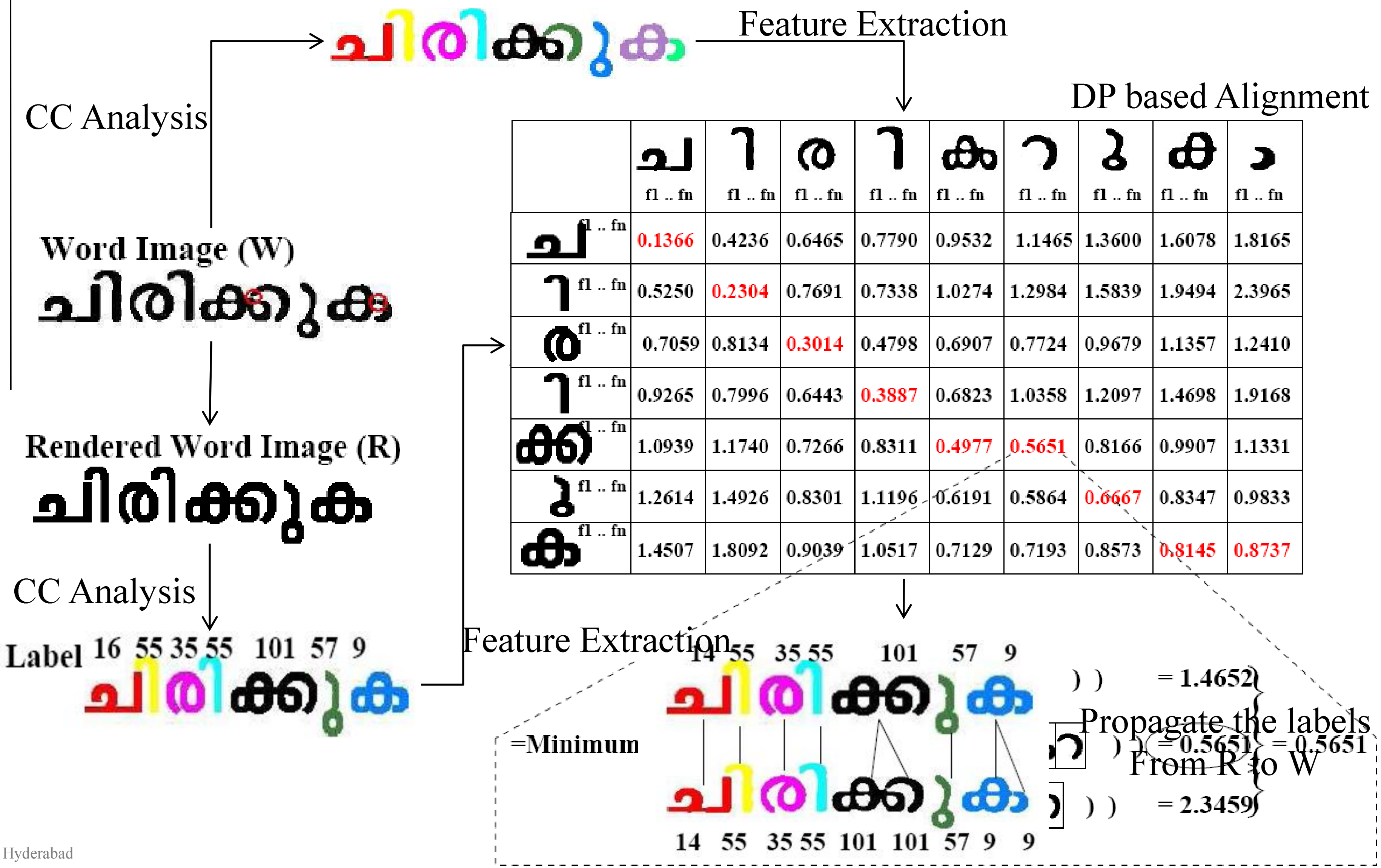


Decision Making Rules in backtracking



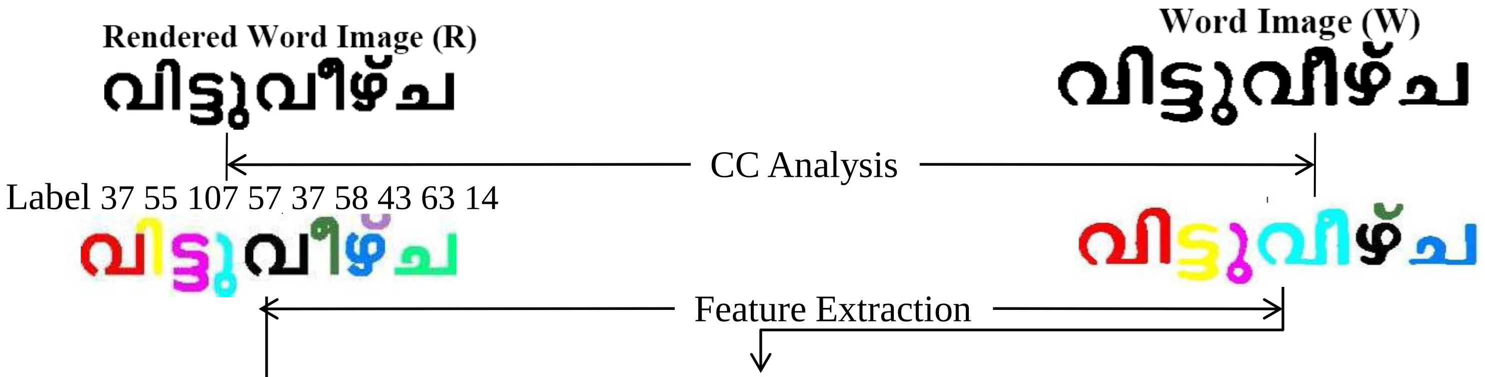
	First	Second	R - 1	R - 2	Condition	Decision1	Decision2
1.	M	I	M1	M2	$M1 < M2$	M, N	CUT
2.	MM	I	M1	M2	$M1 < M2$	MM/DS, N	CUT
3.	M	D	M1	M3	$M1 < M3$	M, MS	MERGE
4.	MM	D	M1	M3	$M1 < M3$	MM/DS, MS	MERGE
5.	M	IM	M1	M2	$M1 < M2$	M, N	CUT
6.	MM	DM	M1	M3	$M1 < M3$	MM/DS, MS	MERGE
7.	I	M	M1	M2	$M1 < M2$	M, N	CUT
8.	I	MM	M1	M2	$M1 < M2$	MM/DS, N	CUT
9.	D	M	M1	M3	$M1 < M3$	M, MS	MERGE
10.	D	MM	M1	M3	$M1 < M3$	MM/DS, MS	MERGE
11.	IM	M	M1	M2	$M1 < M2$	M, N	CUT
12.	DM	MM	M1	M3	$M1 < M3$	MM/DS, MS	MERGE

Alignment of a word with two cuts



	ച f1 .. fn	ി f1 .. fn	ര f1 .. fn	ി f1 .. fn	ക്ക f1 .. fn	൭ f1 .. fn	ു f1 .. fn	ക f1 .. fn	ാ f1 .. fn
ച f1 .. fn	0.1366	0.4236	0.6465	0.7790	0.9532	1.1465	1.3600	1.6078	1.8165
ി f1 .. fn	0.5250	0.2304	0.7691	0.7338	1.0274	1.2984	1.5839	1.9494	2.3965
ര f1 .. fn	0.7059	0.8134	0.3014	0.4798	0.6907	0.7724	0.9679	1.1357	1.2410
ി f1 .. fn	0.9265	0.7996	0.6443	0.3887	0.6823	1.0358	1.2097	1.4698	1.9168
ക്ക f1 .. fn	1.0939	1.1740	0.7266	0.8311	0.4977	0.5651	0.8166	0.9907	1.1331
ു f1 .. fn	1.2614	1.4926	0.8301	1.1196	0.6191	0.5864	0.6667	0.8347	0.9833
ക f1 .. fn	1.4507	1.8092	0.9039	1.0517	0.7129	0.7193	0.8573	0.8145	0.8737

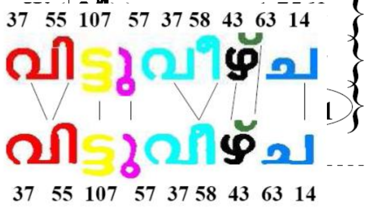
Alignment of a word with two merges



	വി f1 .. fn	ട്ട f1 .. fn	ു f1 .. fn	വീ f1 .. fn	ഴ f1 .. fn	് f1 .. fn	ച f1 .. fn
വ f1 .. fn	0.3660	0.8032	1.1895	1.5318	1.8417	2.1886	2.4068
ി f1 .. fn	0.6557	1.1354	1.3356	1.4622	2.0569	2.6310	2.7503
ട്ട f1 .. fn	0.8579	0.6789	0.7810	0.9603	1.2099	1.3747	1.8787
ു f1 .. fn	1.0520	0.7853	0.7499	1.0164	1.1432	1.3219	1.7282
വ f1 .. fn	1.1666	0.9901	0.9398	0.9264	1.2363	1.5833	1.6614
ീ f1 .. fn	1.4322	1.3780	1.4077	1.0631	1.4904	1.9453	2.0514
ഴ f1 .. fn	1.7886	1.5261	1.5014	1.5310	1.1236	1.2390	2.0424
് f1 .. fn	2.0392	1.7265	1.7031	2.0524	1.3401	1.3361	1.7355
ച f1 .. fn	2.3007	2.2585	2.0051	2.0249	1.8020	1.8824	1.4421

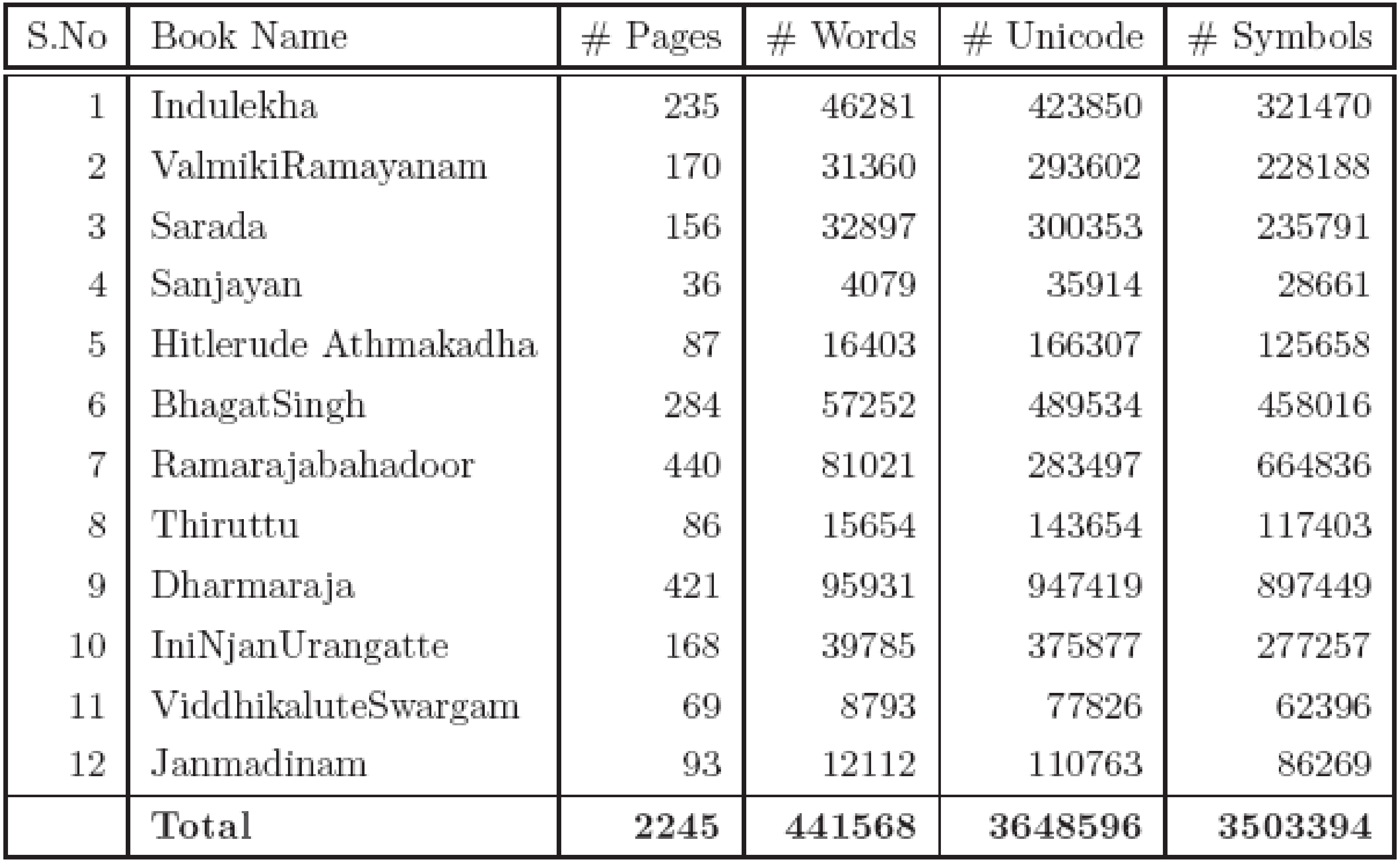
DP based Alignment

= Minimum { **Propagate the labels from R to W**



} = 1.1061

Statistics of Malayalam books used in the experiments



S.No	Book Name	# Pages	# Words	# Unicode	# Symbols
1	Indulekha	235	46281	423850	321470
2	ValmikiRamayanam	170	31360	293602	228188
3	Sarada	156	32897	300353	235791
4	Sanjayan	36	4079	35914	28661
5	Hitlerude Athmakadha	87	16403	166307	125658
6	BhagatSingh	284	57252	489534	458016
7	Ramarajabahadoor	440	81021	283497	664836
8	Thiruttu	86	15654	143654	117403
9	Dharmaraja	421	95931	947419	897449
10	IniNjanUrangatte	168	39785	375877	277257
11	ViddhikaluteSwargam	69	8793	77826	62396
12	Janmadinam	93	12112	110763	86269
	Total	**2245**	**441568**	**3648596**	**3503394**

Unigram and Bigram statistics

- Obtained the Unigram and Bigram statistics in Malayalam as a by-product of word alignment.

S.No	Char	Unigram	S.No	Char	Unigram
1.	ി	0.0812	11.	ം	0.0271
2.	ാ	0.0777	12.	േ	0.0262
3.	ു	0.0746	13.	മ	0.0242
4.	െ	0.0399	14.	്	0.0233
5.	യ	0.0339	15.	പ	0.0211
6.	ര	0.0323	16.	ന്ന	0.0193
7.	ക	0.0305	17.	ട	0.0177
8.	വ	0.0297	18.	ക്ക	0.0172
9.	ന	0.0295	19.	ത്ത	0.0164
10.	ത	0.0292	20.	സ	0.0151

S.No	Char Pair	S.No	Char Pair
1.	ു ം	11.	ത്ത ി
2.	ാ യ	12.	വ ി
3.	യ ി	13.	ത ി
4.	ര ു	14.	ി െ
5.	മ ാ	15.	ന ി
6.	ന്ന ു	16.	ി യ
7.	ു ന്ന	17.	ി ക്ക
8.	ക ാ	18.	ര ി
9.	ക്ക ു	19.	ു െ
10.	യ ു	20.	യ ാ

Empirical Evaluation of Character Classification Schemes

Motivation

- Are the state of the art classifiers suitable/ sufficient to solve Large Class problems ?
  - – Most of the classifiers designed for smaller number of classes.
  - – But a large number of real world problems are large class (in the order of hundreds) in nature.
- Will the character classification problem for Indian languages be solved successfully ?
  - – Large number of classes.
  - – Unavailability of bench-mark datasets.

Focus of the Study

- **Experiment 1** : Comparison of classifiers and features.
- **Experiment 2** : Scalability of classifiers.
- **Experiment 3** : Richness in the feature space.
- **Experiment 4** : Sensitivity of features to degradation.
- **Experiment 5** : Generalization across fonts.
- **Experiment 6** : Applicability across scripts.

Classifiers Used

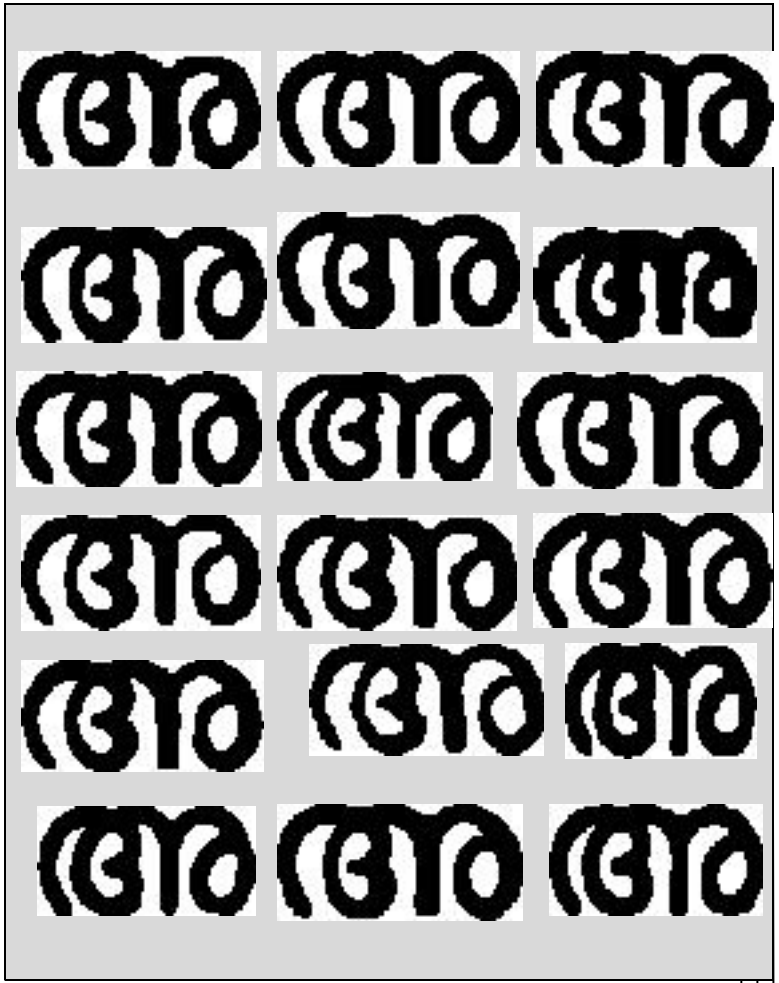
- Multi-layer Perceptron (MLP).
- Convolutional Neural Networks(CNN).
- K-Nearest Neighbour (KNN).
- Approximate Nearest Neighbour(ANN).
- SVM-Majority Voting (SVM-1).
- SVM-DDAG (SVM-2).
- Naive Bayes(NB).
- Decision Tree Classifier (DTC).

Features Used

- Central Moment (CM).
- Zernike Moment (ZM).
- Discrete Cosine Transform(DCT).
- Discrete Fourier Transform(DFT).
- Principal Component analysis(PCA).
- Linear Discriminant Analysis(LDA).
- Random Projections (RP).
- Distance Transform (DT).
- Raw Image (IMG).

Dataset Used

- From annotated books printed primarily in Malayalam.
- 5,00,000 real characters (Symbols) from 5 Books.
- Other scripts used for the experiments are : *Telugu and English.*
- Dataset Generation.



C. V. Jawahar and A. Kumar, .Content-level annotation of large collection of printed document images,. in *ICDAR,* pp. 799.803, 2007.

Comparison of Classifiers and Features:

Experimental Settings

- The focus of the study is to find out the set of classifiers and features that can be used to solve the problem successfully.
- Parameters of classifiers :-
  - MLP – no. of nodes in the hidden layer: 60, momentum: 0.6, no. of epochs: 30.
  - SVM –with linear kernel.
  - KNN and ANN – with K = 5.
- Scale size used : 20 X 20.
- Train : Test Ratio => 5:95

Experiment: 1

Comparison of Classifiers and Features: Results

Feature	Dim	Classifiers						
		MLP	KNN	ANN	SVM-1	SVM-2	NB	DTC
C.M	20	12.04	4.16	5.86	10.04	9.19	11.93	5.57
DFT	16	8.35	8.96	9.35	7.88	7.86	15.33	13.85
DCT	16	5.43	5.11	5.92	5.25	5.24	8.96	7.89
ZM	47	1.30	1.98	2.34	1.24	1.23	3.99	8.04
PCA	350	1.04	1.14	2.39	0.37	0.35	4.83	5.97
LDA	350	0.55	0.52	1.04	0.35	0.34	3.20	4.77
RP	350	0.33	0.50	0.74	0.34	0.34	3.12	8.04
DT	400	1.94	1.27	1.98	1.84	1.84	4.28	2.20
IMG	400	0.32	0.56	0.78	0.32	0.31	1.22	2.45

Error rates on Malayalam dataset.

Error rate using CNN : 0.93

Comparison of Classifiers and Features:

Observations

- SVM classifiers outperforms other classifiers, because of its high generalization capability.
- SVM-2 with a class of feature extraction techniques based on raw images and their projection on uncorrelated set of vectors resulted in the best performance.
- DTC and NB performed the worst of all.
- KNN performed moderately well, but with a higher computational requirement compared to SVM.

Experiment: 2

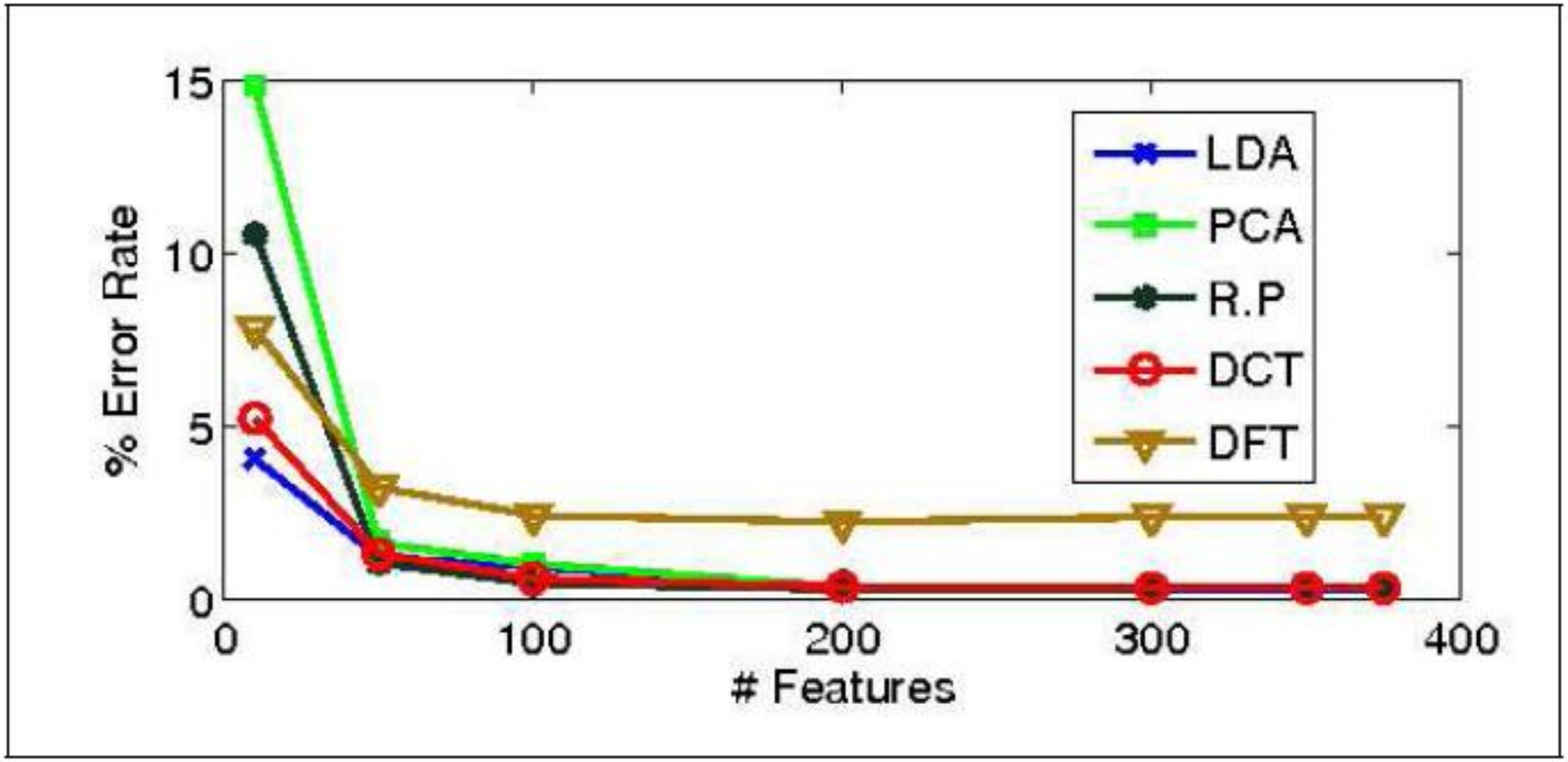
Richness in the Feature Space:

Experimental Settings

- What should be the ideal feature vector length for the problem to get solved successfully ?
- With a large number of features accuracy can be improved.
- We conducted the experiments by varying the feature vector length from 10 to 375.
- Features used for this study are, LDA, PCA, RP, DCT and DFT.

Richness in the Feature Space:

Results



Error rates of SVM-2 classifiers with varying number of features.

Richness in the Feature Space:

Observations

- Error rates rapidly decreases with the increase in number of features initially and then saturates after a point.
- When the number of features are small, LDA outperforms PCA.
- However with a large number of features PCA, LDA, RP performs more or less similarly.
- A rich feature space is needed to solve the classification problem successfully.

Scalability of Classifiers:

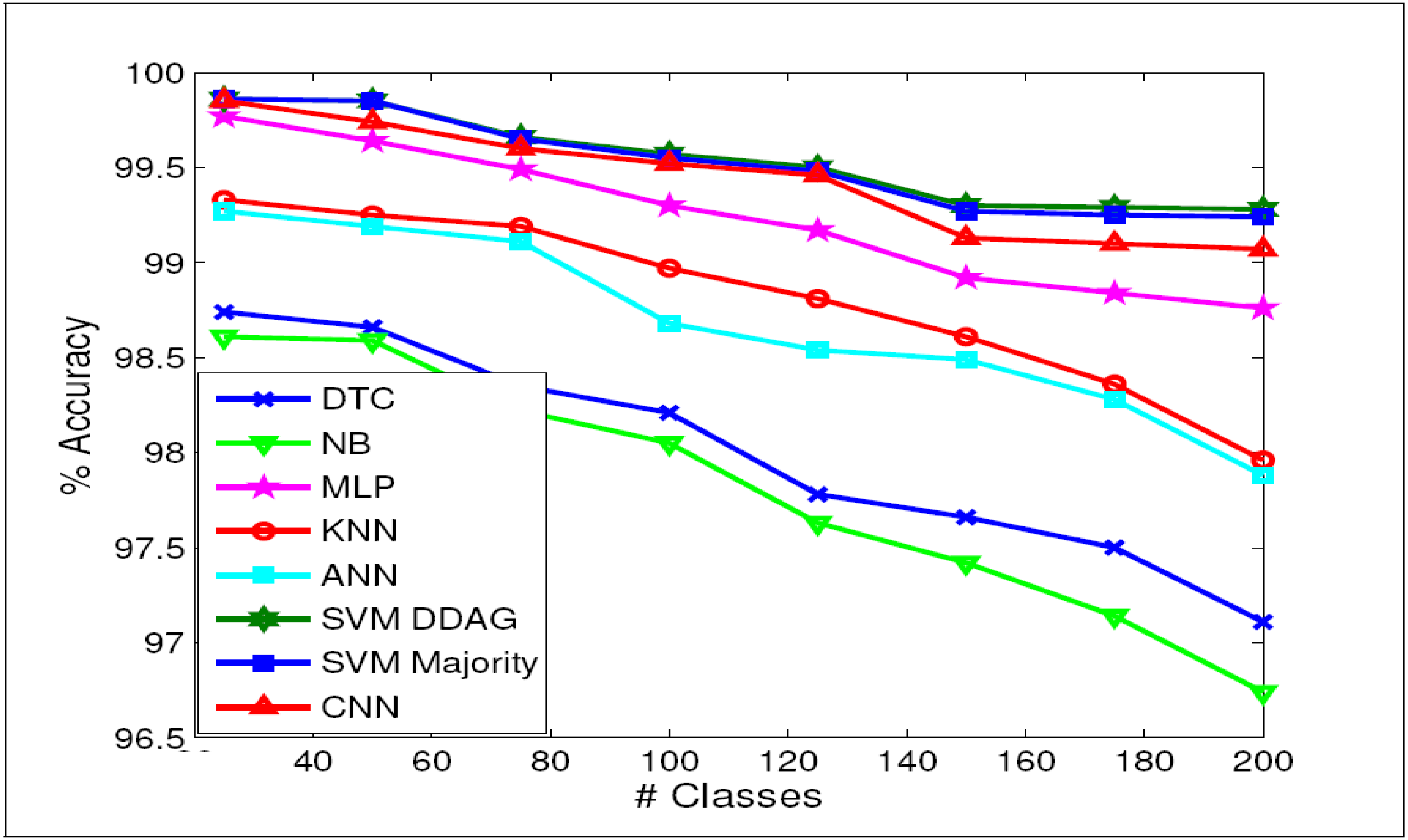
Experimental Settings

- Most of the publicly available datasets have small number of classes (in the order of a few tens).
- One of the major challenges in Indian language character recognition is the large number of classes (in the order of hundreds).
- How the performance of the classifiers effected with the increase in size of the problem (as the number of classes increases)?
- We conducted the experiments by varying the number of classes from 10 to 200.

Experiment: 3

Scalability of Classifiers:

Results



Accuracy of different classifiers Vs no. of classes, Feature used : LDA.

Experiment: 3

Scalability of Classifiers:

Observations

- Performance of all the classifiers goes down as the number of classes increases.
- SVM classifiers degrade gracefully with the increase in size of the problem.

  (For 10 class problem, accuracy = 99.9, For 200 class problem = 99.3).
- The second best performing classifiers are Neural Networks.

Experiment: 4

Degradation of Characters :

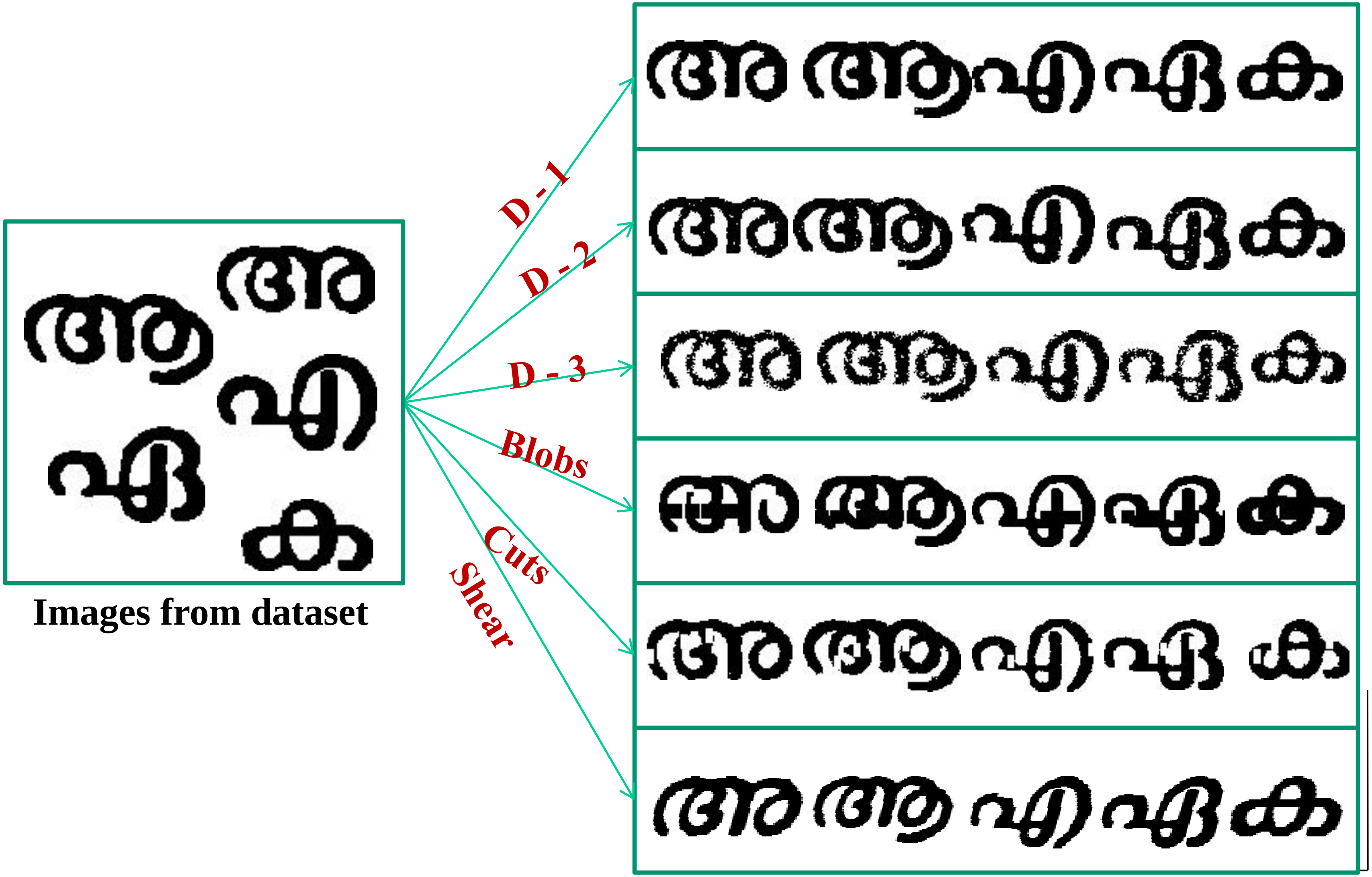
Experimental Settings

- Characters in a real document are generally degraded.
- Most of the feature extraction techniques will have difficulties to extract the right features, in the presence of degradations.
- We modeled 6 degradations.
  - D1, D2, D3 (based on boundary erosions).
  - Cuts, Ink blobs, Shear.

Q. Zheng and T. Kanungo, .Morphological degradation models and their use in document image restoration,. in *ICIP, pp. 193.196, 2001.*

Experiment: 4

Examples of degraded images



Experiment: 4

Degradation of Characters:

Results

Feature	D-1	D-2	D-3	Blobs	Cuts	Shear
C.M	9.45	9.46	10.97	16.28	12.33	30.07
DFT	7.89	7.93	7.98	26.70	8.73	18.90
DCT	5.71	5.72	6.07	19.80	7.93	16.46
ZM	1.96	1.98	2.10	8.41	4.35	17.75
PCA	0.39	0.39	0.40	2.17	0.64	8.59
LDA	0.30	0.31	0.32	2.01	0.61	7.32
RP	0.48	0.67	1.04	3.61	0.71	6.75
DT	1.75	1.98	2.21	10.33	5.07	12.34
IMG	0.32	0.33	0.33	2.78	0.66	6.84

Error rates of different features on various degradations using SVM-2 classifier.

Degradation of Characters :

Observations

- Statistical features are reasonably insensitive to the small degradations (D1, D2 and D3).
- Features like DT, which works well with clean images fails with cuts and ink blobs in the character.
- A better performance is observed for features PCA, LDA, RP and even raw images(IMG) on degradation, compared to others.
- Shear is a more challenging problem, need more consideration in this aspect.

Experiment: 5

Generalization Across Fonts:

Experimental Settings

- How sensitive is the classifier performance on an unseen font ?
- The study included 5 popular fonts in Malayalam.
  - MLTTRevathi, MLTTKarthika, MLTTMalavika, MLTTAmbili, MLTTKaumudi.
- Train the classifier with 4 fonts and test on the 5th font.

Experiment: 5

IIIT Hyderabad

Generalization Across Fonts:

Results and Observations

	Font -1	Font -2	Font -3	Font - 5	Font-4
S1	98.15	95.49	92.52	94.27	92.22
S2	98.97	97.14	95.22	94.59	94.65

Accuracies of SVM-2 classifier when trained with 4 fonts and tested on the 5th font. S1 : Dataset without degradation, S2: Dataset with degradation.

- Generalization across fonts can be achieved by having a wide variety of fonts in the training set.
- A better performance can be achieved by adding a little degradation to the training data.

Applicability across Scripts:

Experimental Settings

- Can the previous experimental results be extended to other scripts ?
- Experiments conducted on Telugu and English scripts.
- Around 50,000 real character images from each scripts used for the experiments.

Experiment: 6

Applicability across Scripts:

Results

Features	Telugu (350 class)		English (72 class)	
	20X20	40X40	20X20	40X40
C.M	20.78	12.32	7.25	6.48
DFT	8.45	5.48	2.04	1.12
DCT	9.67	2.71	2.14	1.04
ZM	15.71	6.71	5.37	3.31
PCA	4.62	2.93	0.86	0.46
LDA	2.56	1.67	0.29	0.23
RP	2.49	1.66	0.28	0.23
DT	3.48	3.17	0.98	0.87
IMG	3.18	2.84	0.28	0.23

Error rates on Telugu and English Datasets, with SVM-2 classifier.

Applicability across Scripts:

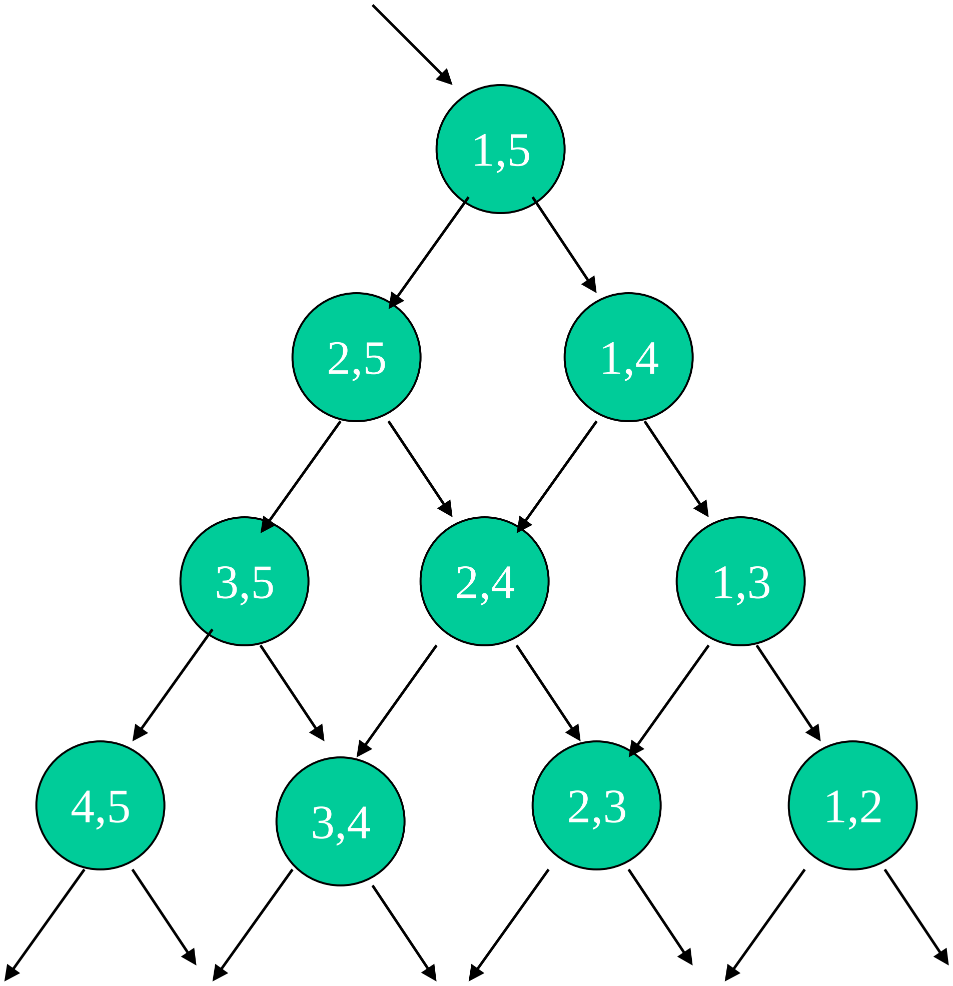
Observations

- The conclusions on character classification are highly script/ language independent.
- More complex scripts can be approached with a richer feature space, which gives more discriminative features.

Design and Efficient Implementation of Classifier for Large Class Problems

- SVMs are popular and accurate binary classifiers with high generalization capability.
- Direct multiclass extension of SVM is not attractive.
- Binary pair-wise classifiers combined using DAG or BHC architectures are generally used.
- For large classes, the solutions are not scalable.
- How to scale SVM solutions for large class classification, in terms of their space and computational requirements?

Decision Directed Acyclic Graph (DDAG)



Properties of Multi-class SVM with DDAG

- Each binary solution is independent.
- Space complexity ~ total number of SVs in the solution.

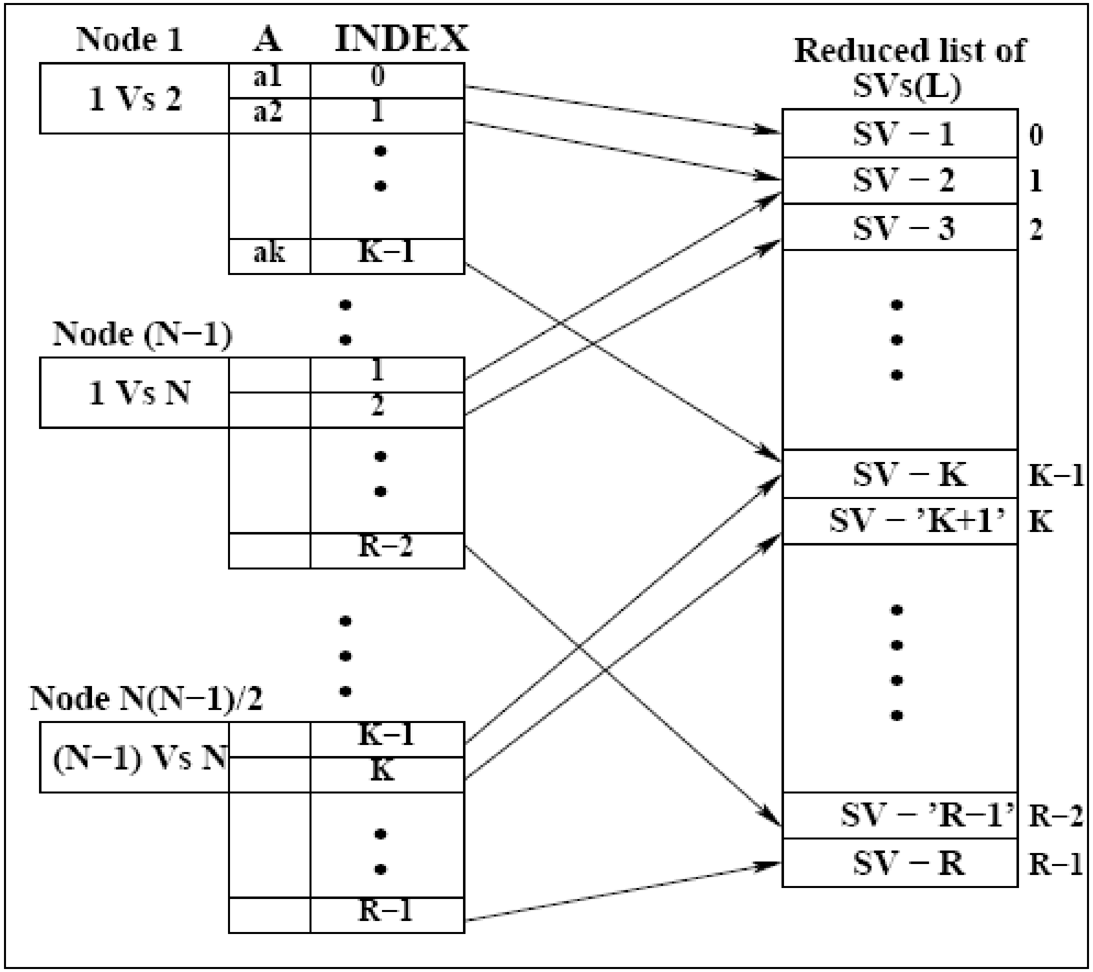
$$f(x) = \sum_{i=1}^{r} \alpha_i y_i K(x, s_i) + b.$$

- Time complexity for classifying a sample ~ number of SVs along the decision path it takes.
- As number of classes increases, the complexities increase multi-fold.
- For large classes, the solution becomes impractical.
- How to reduce the complexities without compromising on the accuracy?

Proposed Solution

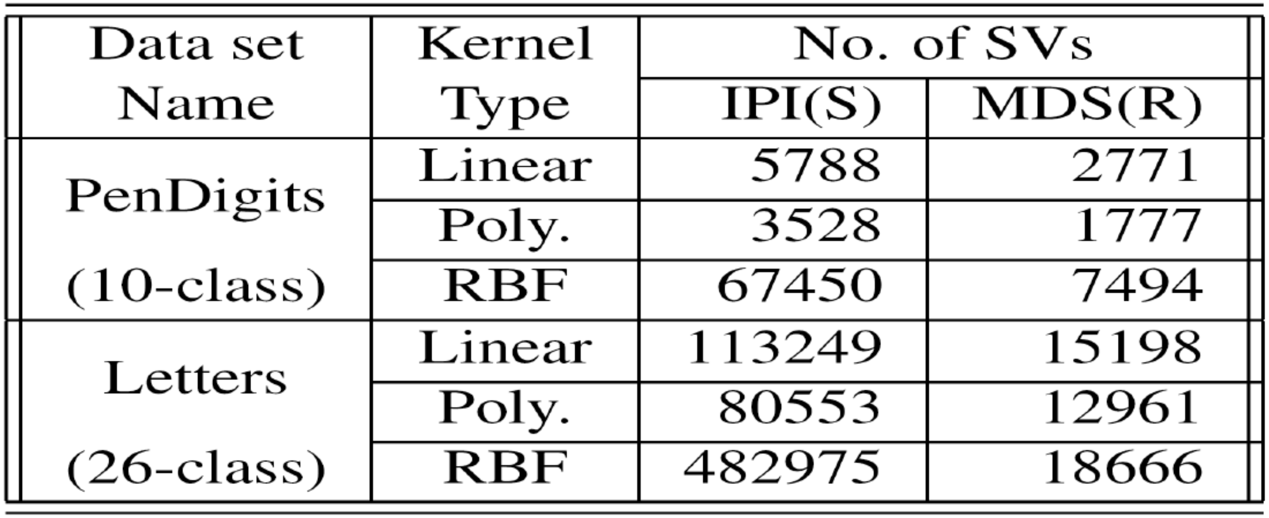
- An effective and easy to implement data structure for efficiently storing SVs.
  - We call it Multiclass Data Structure (MDS).
  - Breaks the independence assumption – SVs are samples on class boundaries.
  - Exploits the redundancies in SVs across the pair-wise classifiers.
  - As number of classes increases, the redundancy also increase.
- An algebraic method for simplifying hierarchical SVM solutions *exactly*.

MDS: Multiclass Data Structure



- **The kernel computation for a SV once computed is *reused in computing at other nodes.***

MDS Vs IPI on UCI datasets



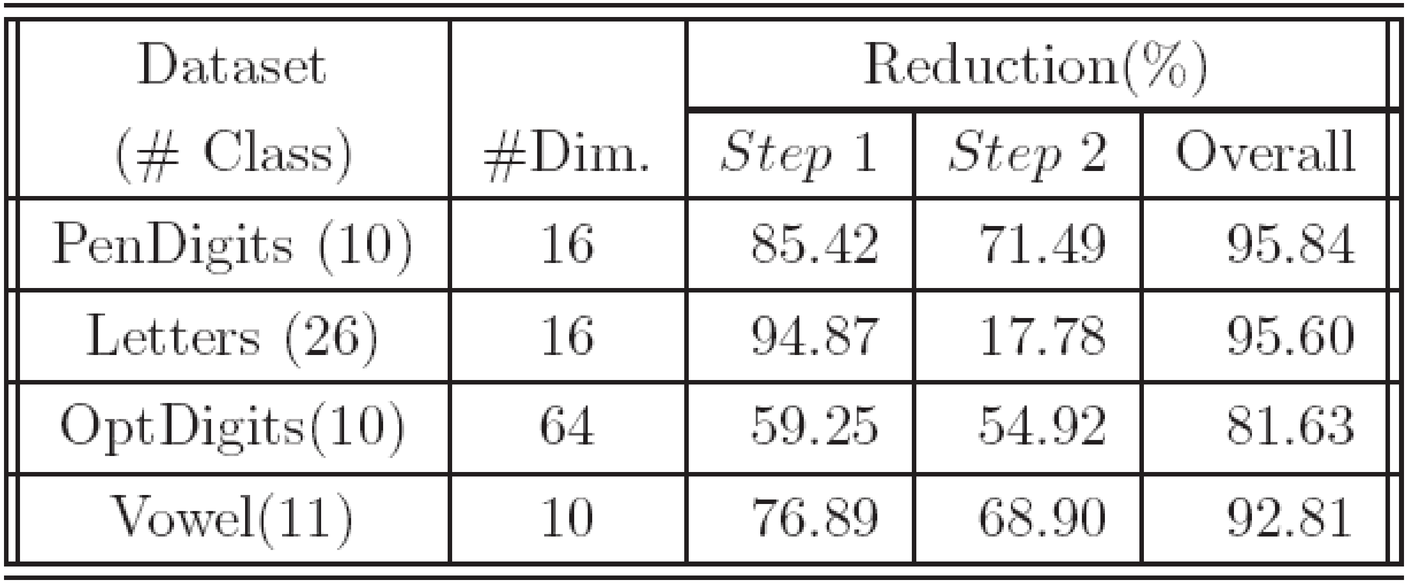
Data set Name	Kernel Type	No. of SVs	
		IPI(S)	MDS(R)
PenDigits (10-class)	Linear	5788	2771
	Poly.	3528	1777
	RBF	67450	7494
Letters (26-class)	Linear	113249	15198
	Poly.	80553	12961
	RBF	482975	18666

- The utility of MDS, increases with the size of the problem.
- With the use of MDS, we achieved 98.5% of reduction in SVs and 60% reduction in classification time .(using linear and polynomial kernels on a 300class data set in comparison to a naïve implementation.)

Algebraic Exact Simplification

- ***Step 1: Multiclass extension***
  - Apply *exact simplification proposed by T. Downs et al. in JMLR, 2001 to each node* independently.
  - Each node is reduced to a set of linearly independent support vectors.
- ***Step 2 : Hierarchical Exact Simplification (HES)***
  - Union of two linearly independent sets need not be independent.
  - Add SVs from nodes that are above in a decision path to the one below.
  - Reduce the obtained extended set by exact simplification method.
  - Apply *HES along each decision path independently.*

Results with Algebraic Exact Simplification

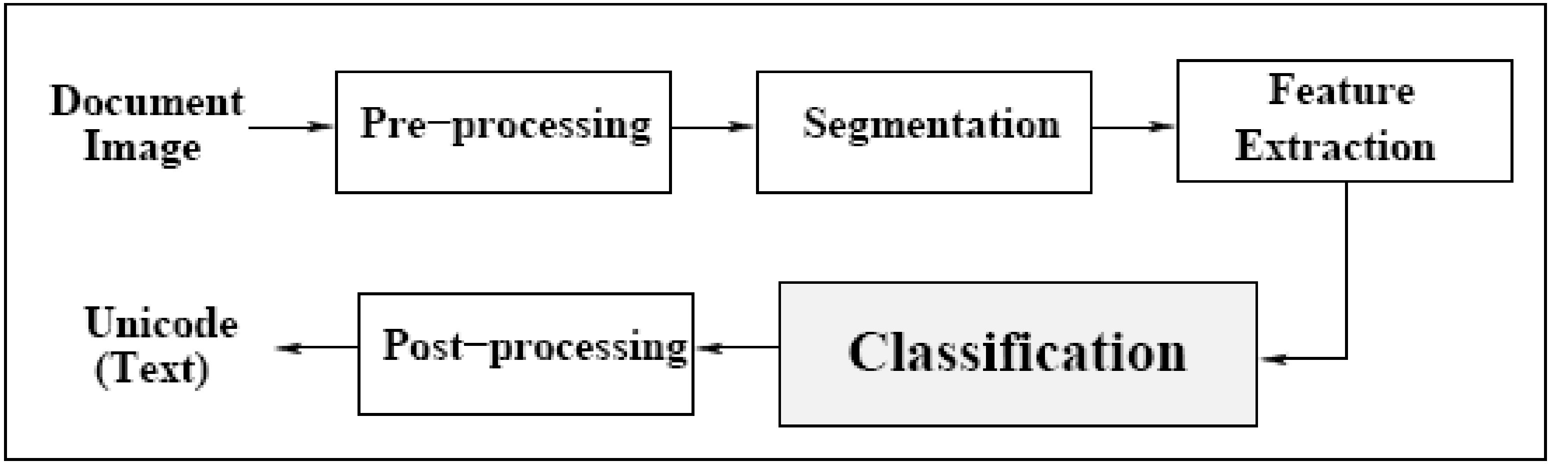


Dataset (# Class)	#Dim.	Reduction(%)		
		Step 1	*Step* 2	Overall
PenDigits (10)	16	85.42	71.49	95.84
Letters (26)	16	94.87	17.78	95.60
OptDigits(10)	64	59.25	54.92	81.63
Vowel(11)	10	76.89	68.90	92.81

Reduction in classification time (using linear kernel).

- With the use of HES the time complexity of multiclass problems can be reduced considerably.

Character Classification for OCR



Our OCR Design Specifications

- Used SVM with DDAG as classifier and binarized Pixel values as feature set.
- The training data for SVM is collected from the real dataset.
- No. of classes : 205
- Tweaked pre-processing routines to fit the Malayalam dataset.

Performance Evaluation

- Performance Metrics

$$\text{Character Edit Distance (CER)} = \frac{\text{CharEditDistance}(C,O)}{|C|}$$

$$\text{Symbol Error Rate} = \frac{\text{No. of Misclassified and Recognizable Symbols}}{\text{Total No. of Recognizable Symbols}}$$

$$\text{Unicode Error Rate} = \frac{\text{No. of Misclassified Unicode}}{\text{Total No. of Unicode}}$$

Symbol level and Unicode level Error Rates

S.No	Book Name	Symbols		Unicode	
		# Total	Error(%)	# Total	Error(%)
1.	Indulekha	321470	1.70	423884	4.80
2.	ValmikiRamayanam	228188	0.94	293822	2.82
3.	Sarada	235791	2.76	299056	3.92
4.	Sanjayan	28661	3.34	35668	4.59
5.	Hitlerude Athmakadha	125658	1.23	163863	2.95
6.	BhagatSingh	458016	3.11	489534	6.39
7.	Ramarajabahadoor	216744	2.12	268653	4.38
8.	Thiruttu	117403	3.88	143582	5.71
9.	Dharmaraja	897449	2.35	947419	5.82
10.	IniNjanUrangatte	277257	1.71	315259	3.74
11.	ViddhikaluteSwargam	62396	3.34	74719	6.89
12.	Janmadinam	86269	2.41	108881	4.98
	Total/ Average	**30,55,302**	**2.40**	**35,64,340**	**4.74**

Unicode level Error Rates

S.No	Book Name	Edit Dist	Substitution	Inserts	Delets
1.	Indulekha	4.80	2.13	2.00	0.98
2.	ValmikiRamayanam	2.82	1.49	0.79	0.65
3.	Sarada	3.92	2.10	1.22	0.89
4.	Sanjayan	4.59	2.31	1.09	1.57
5.	Hitlerude Athmakadha	2.95	1.35	0.70	0.94
6.	BhagatSingh	6.39	3.94	3.18	1.19
7.	Ramarajabahadoor	4.38	2.57	1.95	0.80
8.	Thiruttu	5.71	4.92	2.53	1.26
9.	Dharmaraja	5.82	1.43	1.68	1.30
10.	IniNjanUrangatte	3.74	1.89	0.75	1.19
11.	ViddhikaluteSwargam	6.89	2.45	1.29	2.75
12.	Janmadinam	4.98	1.80	0.82	2.24
	Total/ Average	**4.44**	**2.60**	**1.68**	**1.03**

Word Level Error Rates

- **Accuracy on all the words**

$$\text{Word level Accuracy} = \frac{\text{No. of Correct Words}}{\text{Total No. of Words}}$$

- **Accuracy on recognizable words (Words with no degradations)**

$$\text{Word level Accuracy} = \frac{\text{No. of Correct or Recognizable Words}}{\text{Total No. of Recognizable Words}}$$

Word Level Results

S.No	Book Name	# Total	%Accuracy	# Good	%Accuracy
1.	Indulekha	46281	80.70	39644	94.22
2.	ValmikiRamayanam	31360	90.30	29339	94.22
3.	Sarada	32897	72.10	26671	88.93
4.	Sanjayan	4079	70.63	3224	89.36
5.	Hitlerude Athmakadha	16403	81.32	14291	93.33
6.	BhagatSingh	57252	51.32	35246	80.12
7.	Ramarajabahadoor	81021	53.84	52047	83.81
8.	Thiruttu	15654	59.74	10553	88.61
9.	Dharmaraja	95931	51.83	65427	86.58
10.	IniNjanUrangatte	39785	81.82	34822	93.48
11.	ViddhikaluteSwargam	8793	63.18	6435	86.34
12.	Janmadinam	12112	77.27	10326	90.63
	Total/ Average	**441568**	**69.50**	**328025**	**89.13**

Word Level Results

S.No	Book Name	# Total	0 Error	1 Error	2 Errors
1.	Indulekha	46281	19.29	6.90	4.69
2.	ValmikiRamayanam	31360	9.70	2.71	4.12
3.	Sarada	32897	27.89	9.17	7.75
4.	Sanjayan	4079	29.36	6.47	9.70
5.	Hitlerude Athmakadha	16403	18.67	8.90	5.35
6.	BhagatSingh	57252	48.67	12.42	13.46
7.	Ramarajabahadoor	81021	46.15	8.39	13.12
8.	Thiruttu	15654	40.25	4.22	16.60
9.	Dharmaraja	95931	48.16	13.16	14.96
10.	IniNjanUrangatte	39785	18.17	6.11	6.33
11.	ViddhikaluteSwargam	8793	36.81	8.43	6.96
12.	Janmadinam	12112	22.72	8.99	5.30
	Total/ Average	**441568**	**30.48**	**7.98**	**9.02**

Page Level Results

S.No	Book Name	#Total	Avg.	≤ 2%	2 − 5%	5 − 10%	> 10%
1.	Indulekha	235	4.80	0.42	70.63	26.38	2.55
2.	ValmikiRamayanam	170	2.82	31.76	57.64	4.11	6.47
3.	Sarada	156	3.92	6.41	75.00	14.74	3.20
4.	Sanjayan	36	4.59	5.55	55.55	30.55	5.55
5.	Hitlerude Athmakadha	87	2.94	3.44	89.65	2.29	3.44
6.	BhagatSingh	284	7.39	0	29.92	52.46	16.90
7.	Ramarajabahadoor	440	4.38	7.80	68.79	20.56	2.83
8.	Thiruttu	86	5.71	2.32	52.32	30.23	15.11
9.	Dharmaraja	421	5.85	0.23	56.53	33.96	8.78
10.	IniNjanUrangatte	168	3.74	4.61	66.15	11.28	1.02
11.	ViddhikaluteSwargam	69	6.73	0	33.33	57.97	7.24
12.	Janmadinam	93	4.97	0	69.89	20.43	9.67
	Total/ Average	**2245**	**4.82**	**5.21**	**60.45**	**25.41**	**6.89**

Summary of Results on 12 Malayalam Books

	Total	Error Rate
Symbols	30,55,302	2.40
Unicode	35,64,340	4.74
All Words	4,41,568	30.50
Recognizable Words	3,28,025	10.87
Page Level	2,245	4.82

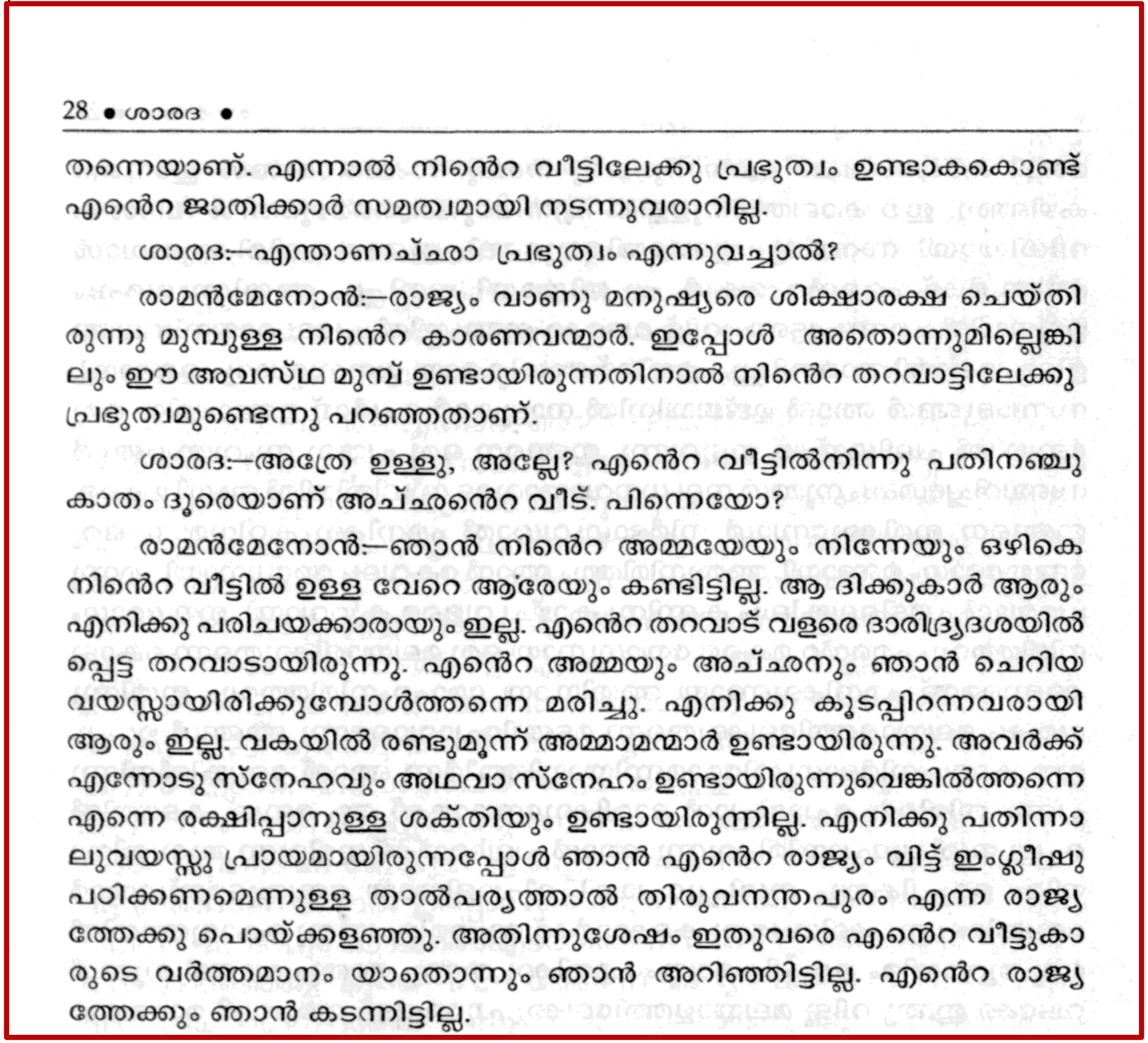
Comparison with Nayana

S.No	Book Name	No. of Pages	Edit Distance		Substitution Err.	
			Nayana	Ours	Nayana	Ours
1.	Indulekha	10	13.55	2.32	5.62	1.08
2.	ValmikiRamayanam	7	13.03	2.04	4.28	0.97
3.	Sanjayan	10	34.48	2.76	19.18	1.03
4.	Hitlerude Athmakadha	8	13.96	2.66	4.9	1.05
5.	BhagatSingh	6	9.72	2.63	3.97	1.31
6.	Ramarajabahadoor	10	12.19	3.02	4.05	1.41
7.	Thiruttu	10	13.44	3.91	6.17	2.11
8.	Dharmaraja	9	10.33	2.52	4.19	1.31
9.	IniNjanUrangatte	9	9.49	1.91	4.23	1.02
10.	ViddhikaluteSwargam	3	13.73	4.72	4.86	1.51
11.	Janmadinam	5	15.98	3.89	5.2	1.47
	Total/ Average	**87**	**15.34**	**2.81**	**6.64**	**1.28**

Results on Scanned Quality A documents

Font Name	8		10		12		14	
Amibili	2.00	0.90	1.92	0.88	2.71	0.52	4.99	0.82
Karthika	2.87	1.52	0.70	0.28	1.10	0.44	0.73	0.45
Lohit	3.86	1.29	2.27	1.11	2.51	1.56	3.29	1.75
Nila	2.43	1.22	1.53	0.54	1.54	0.59	2.01	0.60
Revathi	2.57	1.12	0.57	0.24	1.08	0.53	0.74	0.30

Results: Example 1



28 • ശാരദ •

തന്നെയാണ്. എന്നാൽ നിന്റെ വീട്ടിലേക്കു പ്രഭുത്വം ഉണ്ടാകകൊണ്ട് എന്റെ ജാതിക്കാർ സമത്വമായി നടന്നുവരാറില്ല.

ശാരദ:—എന്താണച്ഛാ പ്രഭുത്വം എന്നുവച്ചാൽ?

രാമൻമേനോൻ:—രാജ്യം വാണു മനുഷ്യരെ ശിക്ഷാരക്ഷ ചെയ്തിരുന്നു മുമ്പുള്ള നിന്റെ കാരണവന്മാർ. ഇപ്പോൾ അതൊന്നുമില്ലെങ്കിലും ഈ അവസ്ഥ മുമ്പ് ഉണ്ടായിരുന്നതിനാൽ നിന്റെ തറവാട്ടിലേക്കു പ്രഭുത്വമുണ്ടെന്നു പറഞ്ഞതാണ്.

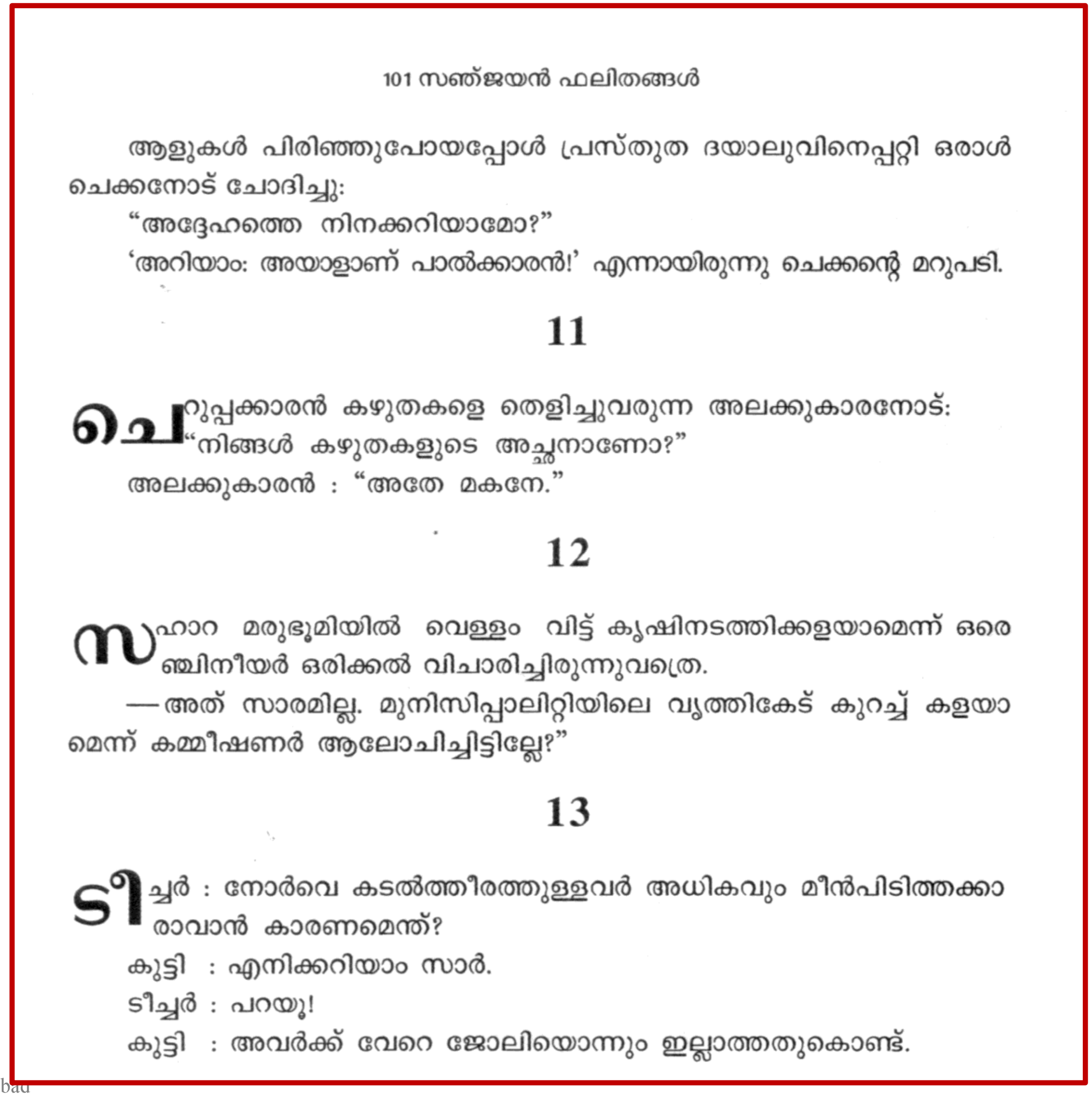
ശാരദ:—അത്രേ ഉള്ളു, അല്ലേ? എന്റെ വീട്ടിൽനിന്നു പതിനഞ്ചു കാതം ദൂരെയാണ് അച്ഛന്റെ വീട്. പിന്നെയോ?

രാമൻമേനോൻ:—ഞാൻ നിന്റെ അമ്മയേയും നിന്നേയും ഒഴികെ നിന്റെ വീട്ടിൽ ഉള്ള വേറെ ആരേയും കണ്ടിട്ടില്ല. ആ ദിക്കുകാർ ആരും എനിക്കു പരിചയക്കാരായും ഇല്ല. എന്റെ തറവാട് വളരെ ദാരിദ്ര്യദശയിൽപ്പെട്ട തറവാടായിരുന്നു. എന്റെ അമ്മയും അച്ഛനും ഞാൻ ചെറിയ വയസ്സായിരിക്കുമ്പോൾത്തന്നെ മരിച്ചു. എനിക്കു കൂടപ്പിറന്നവരായി ആരും ഇല്ല. വകയിൽ രണ്ടുമൂന്ന് അമ്മാമന്മാർ ഉണ്ടായിരുന്നു. അവർക്ക് എന്നോടു സ്നേഹവും അഥവാ സ്നേഹം ഉണ്ടായിരുന്നുവെങ്കിൽത്തന്നെ എന്നെ രക്ഷിപ്പാനുള്ള ശക്തിയും ഉണ്ടായിരുന്നില്ല. എനിക്കു പതിന്നാലുവയസ്സു പ്രായമായിരുന്നപ്പോൾ ഞാൻ എന്റെ രാജ്യം വിട്ട് ഇംഗ്ലീഷു പഠിക്കണമെന്നുള്ള താൽപര്യത്താൽ തിരുവനന്തപുരം എന്ന രാജ്യത്തേക്കു പൊയ്ക്കളഞ്ഞു. അതിന്നുശേഷം ഇതുവരെ എന്റെ വീട്ടുകാരുടെ വർത്തമാനം യാതൊന്നും ഞാൻ അറിഞ്ഞിട്ടില്ല. എന്റെ രാജ്യത്തേക്കും ഞാൻ കടന്നിട്ടില്ല.

Book name : ***Sarada***

Error rate : 5.61%

Results: Example 2



101 സഞ്ജയൻ ഫലിതങ്ങൾ

ആളുകൾ പിരിഞ്ഞുപോയപ്പോൾ പ്രസ്തുത ദയാലുവിനെപ്പറ്റി ഒരാൾ ചെക്കനോട് ചോദിച്ചു:

“അദ്ദേഹത്തെ നിനക്കറിയാമോ?”

‘അറിയാം: അയാളാണ് പാൽക്കാരൻ!’ എന്നായിരുന്നു ചെക്കന്റെ മറുപടി.

11

ചെറുപ്പക്കാരൻ കഴുതകളെ തെളിച്ചുവരുന്ന അലക്കുകാരനോട്: “നിങ്ങൾ കഴുതകളുടെ അച്ഛനാണോ?”

അലക്കുകാരൻ : “അതേ മകനേ.”

12

സഹാറ മരുഭൂമിയിൽ വെള്ളം വിട്ട് കൃഷിനടത്തിക്കളയാമെന്ന് ഒരെഞ്ചിനീയർ ഒരിക്കൽ വിചാരിച്ചിരുന്നുവത്രെ.

—അത് സാരമില്ല. മുനിസിപ്പാലിറ്റിയിലെ വൃത്തികേട് കുറച്ച് കളയാമെന്ന് കമ്മീഷണർ ആലോചിച്ചിട്ടില്ലേ?”

13

ടീച്ചർ : നോർവെ കടൽത്തീരത്തുള്ളവർ അധികവും മീൻപിടിത്തക്കാരാവാൻ കാരണമെന്ത്?

കുട്ടി : എനിക്കറിയാം സാർ.

ടീച്ചർ : പറയൂ!

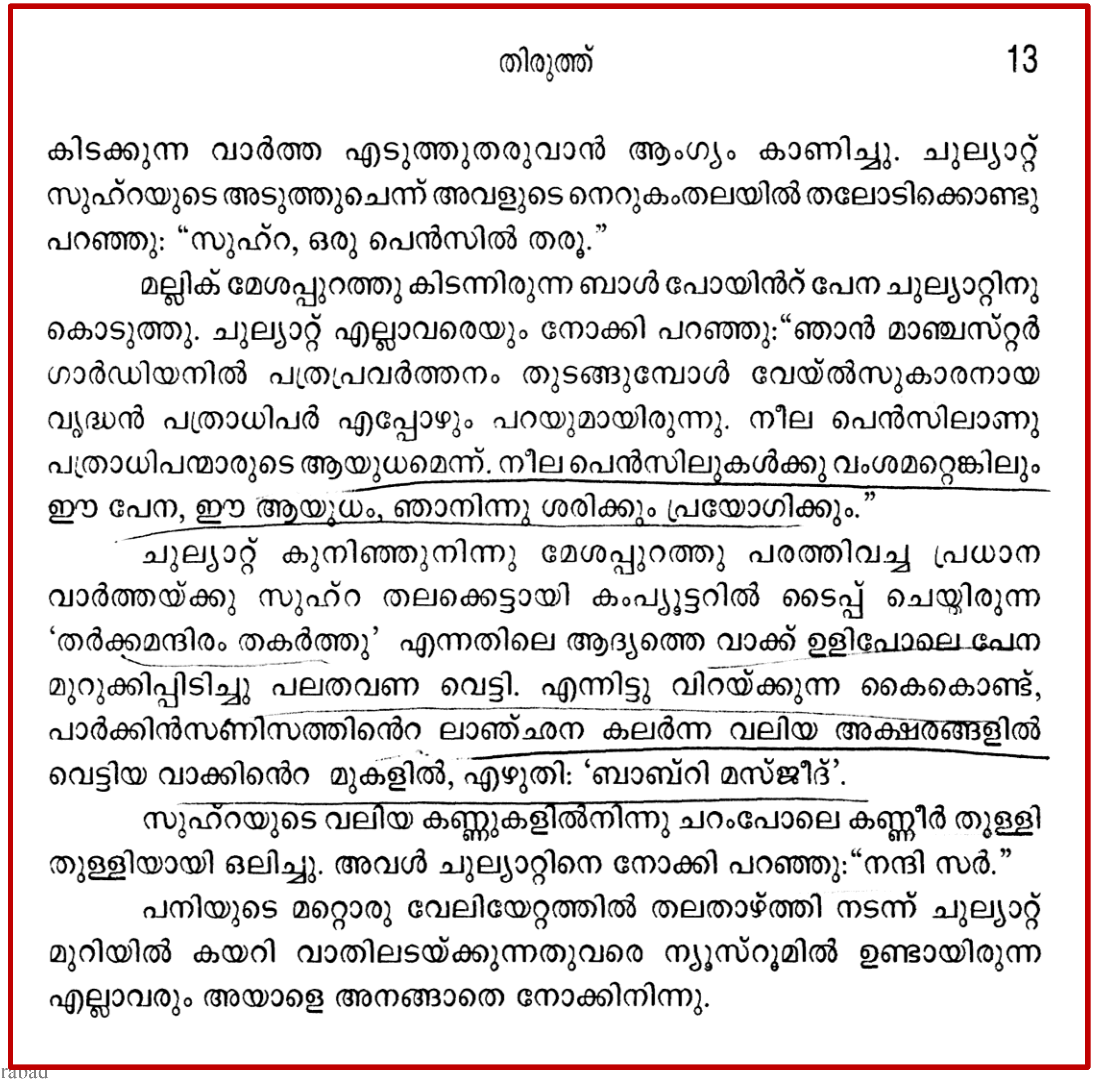
കുട്ടി : അവർക്ക് വേറെ ജോലിയൊന്നും ഇല്ലാത്തതുകൊണ്ട്.

Book name : ***Sanjayan***

Error rate : 26.84%

Sub. Error : 10.85%

Results: Example 3



തിരുത്ത് 13

കിടക്കുന്ന വാർത്ത എടുത്തുതരുവാൻ ആംഗ്യം കാണിച്ചു. ചുല്യാറ്റ് സുഹ്റയുടെ അടുത്തുചെന്ന് അവളുടെ നെറുകംതലയിൽ തലോടിക്കൊണ്ടു പറഞ്ഞു: “സുഹ്റ, ഒരു പെൻസിൽ തരൂ.”

മല്ലിക് മേശപ്പുറത്തു കിടന്നിരുന്ന ബാൾ പോയിൻറ് പേന ചുല്യാറ്റിനു കൊടുത്തു. ചുല്യാറ്റ് എല്ലാവരെയും നോക്കി പറഞ്ഞു:“ഞാൻ മാഞ്ചസ്റ്റർ ഗാർഡിയനിൽ പത്രപ്രവർത്തനം തുടങ്ങുമ്പോൾ വേയ്ൽസുകാരനായ വൃദ്ധൻ പത്രാധിപർ എപ്പോഴും പറയുമായിരുന്നു. നീല പെൻസിലാണു പത്രാധിപന്മാരുടെ ആയുധമെന്ന്. നീല പെൻസിലുകൾക്കു വംശമറ്റെങ്കിലും ഈ പേന, ഈ ആയുധം, ഞാനിന്നു ശരിക്കും പ്രയോഗിക്കും.”

ചുല്യാറ്റ് കുനിഞ്ഞുനിന്നു മേശപ്പുറത്തു പരത്തിവച്ച പ്രധാന വാർത്തയ്ക്കു സുഹ്റ തലക്കെട്ടായി കംപ്യൂട്ടറിൽ ടൈപ്പ് ചെയ്തിരുന്ന ‘തർക്കമന്ദിരം തകർത്തു’ എന്നതിലെ ആദ്യത്തെ വാക്ക് ഉളിപോലെ പേന മുറുക്കിപ്പിടിച്ചു പലതവണ വെട്ടി. എന്നിട്ടു വിറയ്ക്കുന്ന കൈകൊണ്ട്, പാർക്കിൻസണിസത്തിന്റെ ലാഞ്ഛന കലർന്ന വലിയ അക്ഷരങ്ങളിൽ വെട്ടിയ വാക്കിന്റെ മുകളിൽ, എഴുതി: ‘ബാബ്റി മസ്ജീദ്’.

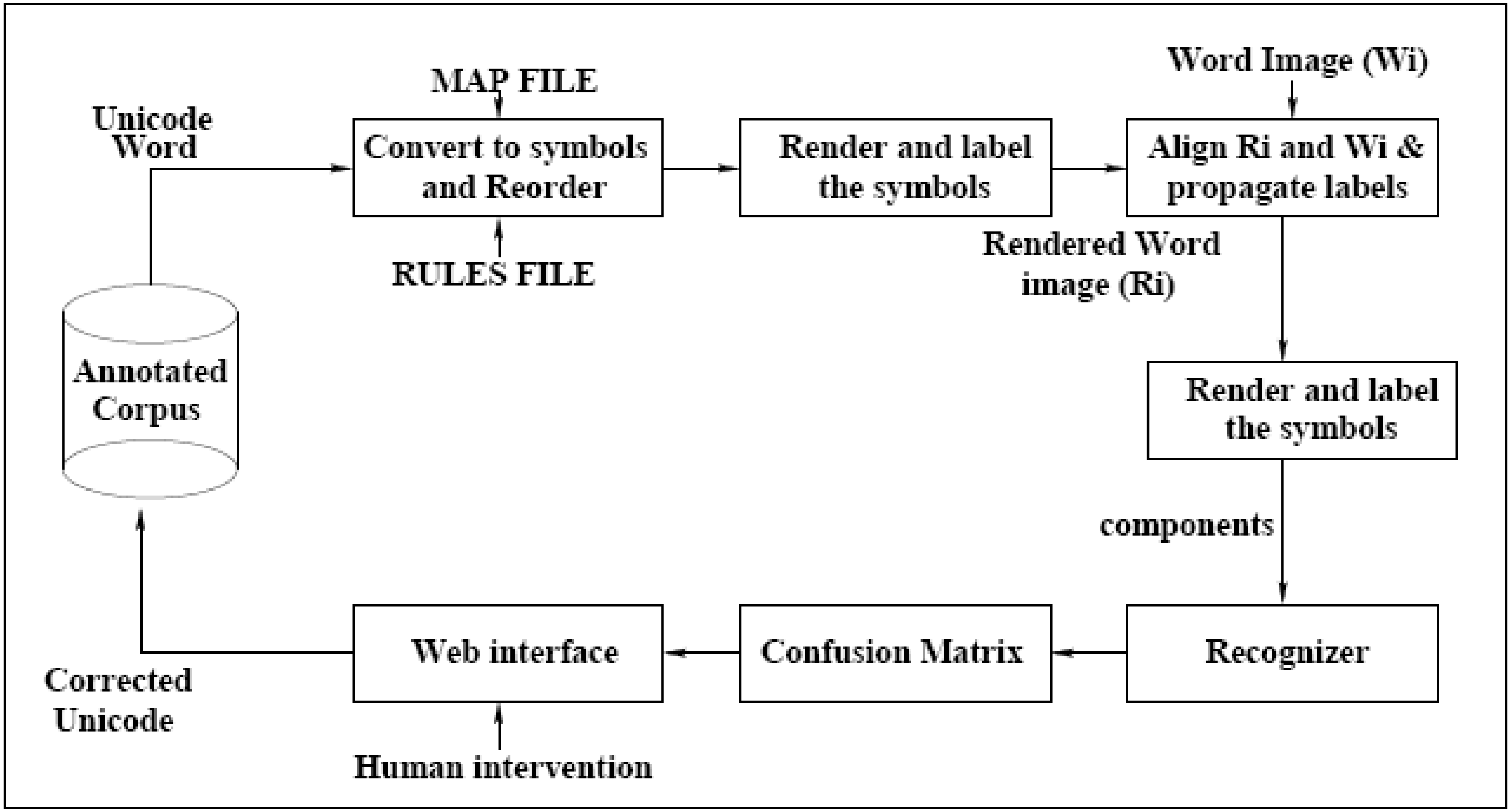
സുഹ്റയുടെ വലിയ കണ്ണുകളിൽനിന്നു ചറംപോലെ കണ്ണീർ തുള്ളി തുള്ളിയായി ഒലിച്ചു. അവൾ ചുല്യാറ്റിനെ നോക്കി പറഞ്ഞു:“നന്ദി സർ.”

പനിയുടെ മറ്റൊരു വേലിയേറ്റത്തിൽ തലതാഴ്ത്തി നടന്ന് ചുല്യാറ്റ് മുറിയിൽ കയറി വാതിലടയ്ക്കുന്നതുവരെ ന്യൂസ്റൂമിൽ ഉണ്ടായിരുന്ന എല്ലാവരും അയാളെ അനങ്ങാതെ നോക്കിനിന്നു.

Book name : ***Thiruttu***

Error rate : 26.84%
Sub. Error : 10.85%

Annotation correction



Procedure for annotation correction with the help of Recognizer.

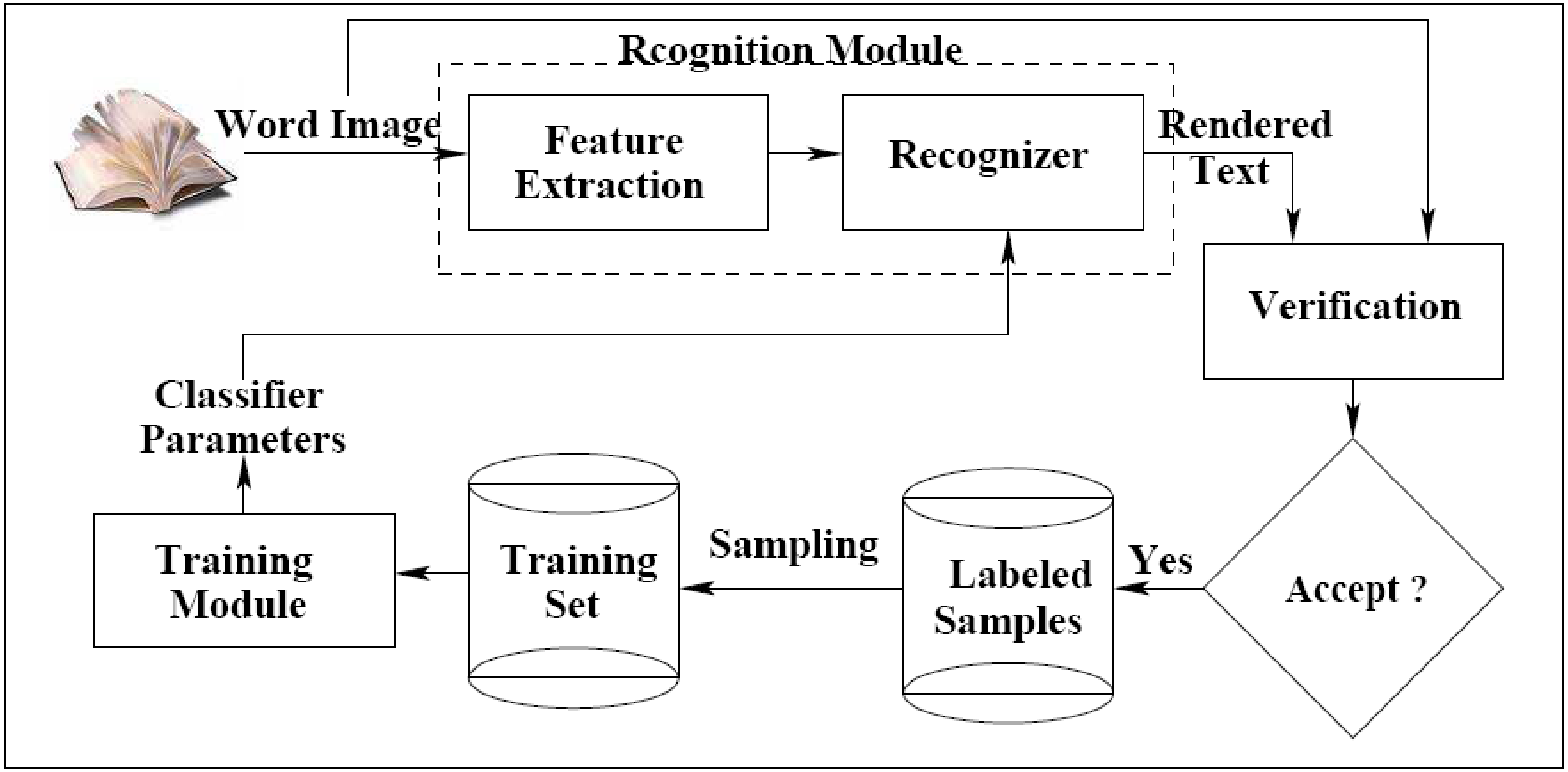
Recognition of Books by Verification and Retraining

- Digital libraries need the recognition of complete book.
- Books are typeset in the same font and style.
- We can use the samples from first few pages to learn the classifier, and obtain better performance over the rest of the collection.
- An automatic learning framework to improve the performance of the classifier over iteration.
- With the help of a high performance *verification module, which acts as a postprocessor* in the system.
- The *Verification module collects / verifies the* samples from the book, labels and stores them.

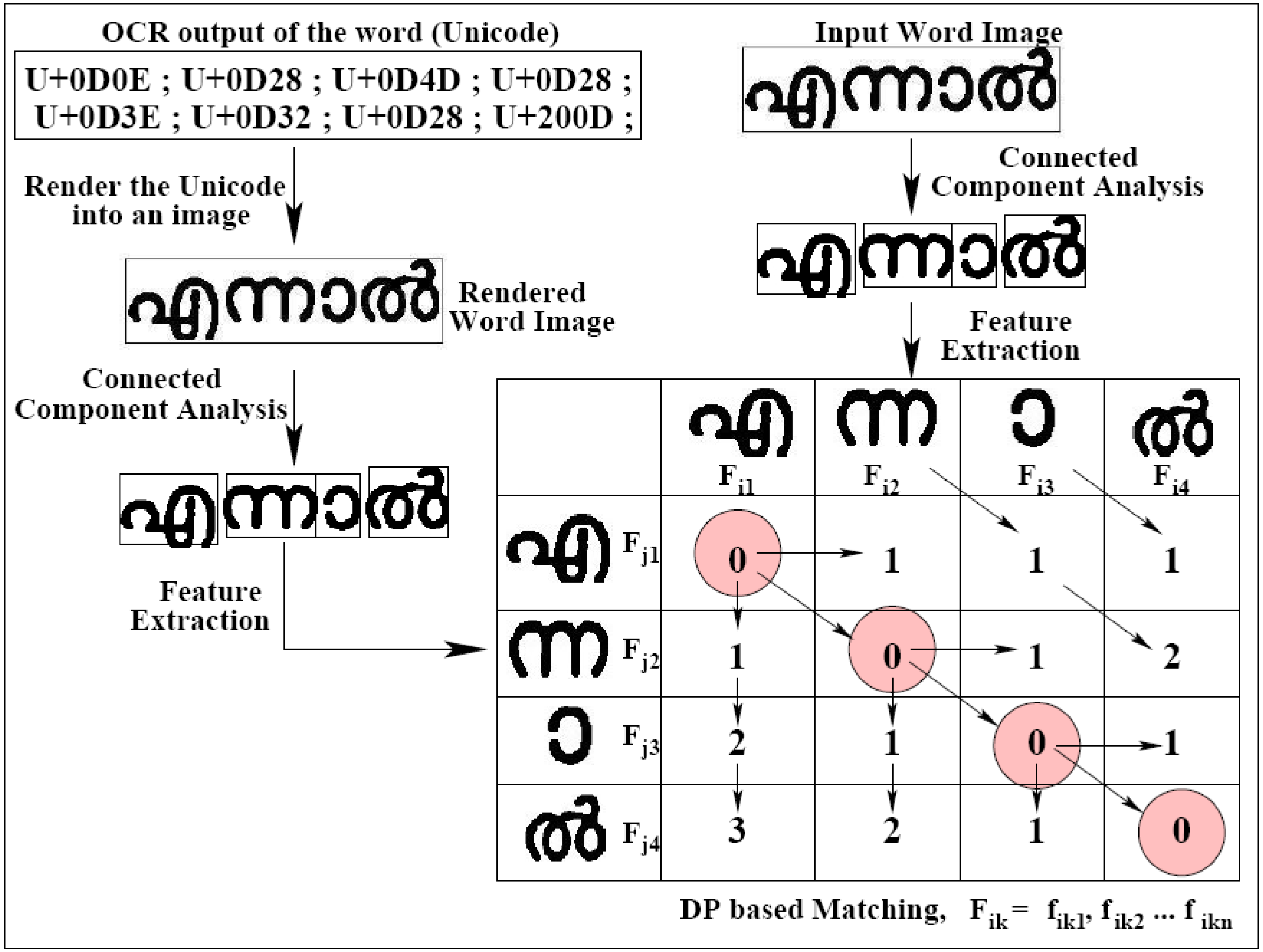
Comparison between existing and proposed system

Traditional OCRs.	***Book Recognizer***
•Designed for isolated pages.	•Designed specifically for books/ large collection of documents.
•Performance remains same over time – the classifier repeats the same mistake.	• Adapt the classifier to the font and style of the collection and improve the performance over time.
•Training is offline (apriori done) – no scope for improvement in performance.	•*New training data is introduced into the* system without any manual intervention.

Overview of the Book Recognition Scheme



An Example for DP based Verification



OCR output of the word (Unicode)

U+0D0E ; U+0D28 ; U+0D4D ; U+0D28 ;
U+0D3E ; U+0D32 ; U+0D28 ; U+200D ;

Render the Unicode into an image

Rendered Word Image

Connected Component Analysis

Feature Extraction

Input Word Image

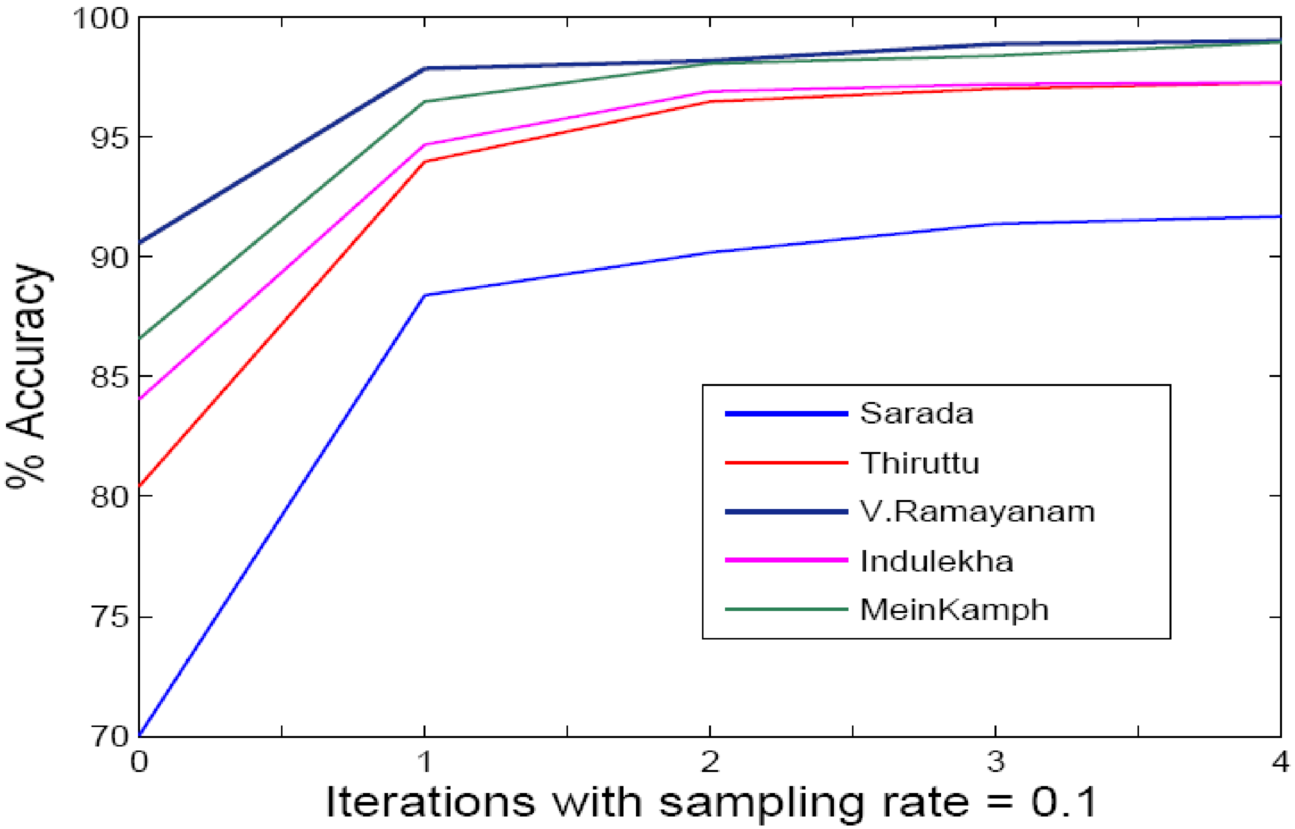
Connected Component Analysis

Feature Extraction

	എ F_{i1}	ന്ന F_{i2}	ാ F_{i3}	ൽ F_{i4}
എ F_{j1}	0	1	1	1
ന്ന F_{j2}	1	0	1	2
ാ F_{j3}	2	1	0	1
ൽ F_{j4}	3	2	1	0

DP based Matching, $F_{ik} = f_{ik1}, f_{ik2} \ldots f_{ikn}$

Improvement in Recognition of books



- We obtain an average performance improvement of 14% in classification accuracies.

Improvement in the Performance of a Book, with varying Sampling Rate.

# Iteration	**Sampling Rate**			
	0.01	**0.05**	**0.1**	**0.3**
0	80.42	80.42	80.42	80.42
1	93.28	94.33	93.96	93.94
2	96.46	96.80	96.47	95.35
3	97.41	97.59	97.01	96.28
4	97.52	97.69	97.26	96.46

- The change in learning rate with the change in the sampling rate is marginal.

Summary of the Work

- The major contributions of the work are :
  - Large dataset generation.
  - Approaches to solve large class problems.
  - Performance evaluation on a huge dataset (Malayalam books).
- We also extend our classifier to continuously improve the performance by providing feedback and retraining the classifier.

Future Scope

- Extend the features and classifier to support more fonts.
- More script specific techniques at pre-processing stage.
- A strong post-processor based on language models. Also, a strong word recognizer as
- a part of post-processor will improve the system.
- Degradation handing: To handle the spurious noise, cuts and merges in the characters.

Thank you ☺

Questions?