உத்தமம்
INFITT
உலகத் தமிழ்த் தகவல் தொழில்நுட்ப மன்றம்
International Forum for Information Technology in Tamil

Penn
University of Pennsylvania

மாநாட்டுக் கட்டுரைகள்
CONFERENCE PAPERS

TAMIL INTERNET 2011

தமிழ் இணையம் 2011

Articificial Intelligence

(செயற்கைத் திறனாய்வு)

Tamil Video Retrieval Based on Categorization in Cloud

V.Akila, Dr.T.Mala
Department of Information Science and Technology,
College of Engineering, Guindy,
Anna University, Chennai
veeakila@gmail.com, malanehru@annauniv.edu

Abstract

Tamil Video retrieval based on categorization in cloud has become a challenging and important issue. Video contains several types of visual information which are difficult to extract in common information retrieval process. Tamil Video retrieval for query clip is a high computation task because of the computation complexity and large amount of data. With cloud computing infrastructure, video retrieval process has some scope and is flexible for deployment. The proposed method categorize the Tamil video into subcategories, splits the video into a sequence of shots and extracts a small number of representative frames from each shot and subsequently calculates frame descriptors depending on the edge and color features. The color histogram is computed for all the key frames based on hue, saturation and intensity values. Edge features are extracted using canny edge detector algorithm. The features extracted are stored in feature library in cloud. The features are tagged with Tamil text in cloud in order to satisfy Tamil query clip. Also, Videos are retrieved based on the Tamil audio information. The EUCALYPTUS cloud computing environment is setup within academic settings and the similarity matching of the Tamil video query is performed. The similar videos are displayed based on the similarity value and the performance is evaluated. Eucalyptus cloud platform is setup in Linux OS and the Tamil video retrieval process is deployed within the cloud. The efficiency of cloud computing technology improves the Tamil video retrieval process and increases the performance.

Keywords—video retrieval, categorization, cloud computing, Tamil query, Eucalyptus

1. Introduction

The need for intelligent processing and analysis of multimedia information has been increasing on a regular basis.

Researchers have found numerous technologies for intelligent video management which includes the shot transition detection, key frame extraction, video retrieval and more. Content based retrieval is considered to be the most difficult and significant issue of practical value amongst all the others. It assists the users in the retrieval of favored video segments from a vast video database efficiently based on the video contents. This paper aims at presenting the process of Tamil video retrieval in cloud environment. Video contains both visual and audio information. Audio contains natural language information which can be used to retrieve similar video content. The Tamil text processing is performed for user Tamil query.

The video retrieval system can be divided into two principal constituents: a module for the extraction of representative characteristics from video segments and defining a retrieval process to find similar

video clips from video database. A large number of approaches use a wide variety of features to symbolize a video sequence of which color histogram; shape information and text analysis are a renowned few. Application that requires a large number of computational resources might have to contact several different resource providers in order to satisfy its requirements. Cloud computing systems provide a wide variety of interfaces ranging from the ability to dynamically provision entire virtual machines. The feature database is stored in the cloud and the users query is compared. As based on cloud computing infrastructure, video retrieval process can be easily extended.

The rest of this paper is organized as follows: Section 2 deals with literature survey in the domain related to the project. It gives the different techniques adopted in the domain. Section 3 deals with system architecture. It includes detailed design of various phases involved in the project. It describes the internal working of the system. Section 4 deals with simulation and results of video retrieval process in cloud for Tamil videos. Section 5 deals with performance evaluation and result analysis. Section 6 focuses on conclusion and future enhancement.

2. Related Works

Nurmi describes the basic principles of the EUCALYPTUS design, that allow cloud to be portable, modular and simple to use on infrastructure commonly found within academic settings [3]. EUCALYPTUS is an open source software framework for cloud computing that implements what is commonly referred to as Infrastructure as a Service. It allow users the ability to run and control entire virtual machine instances deployed across a variety physical resources.

Takagi explains a method for video categorization based on the camera motion[5].Camera motion parameters in the video sequence contain very significant information for categorization of video, because in most of the video, camera motions are closely related to the actions taken. Camera motion parameters can be extracted from video sequence by analyzing motion information. Camera motion parameter has many advantages for categorization of video. Camera motion parameters like pan, fix are obtained using motion vector. Motion vectors are classified into 8 directions and histogram is calculated in each category. By analyzing characteristics of this histogram, camera motion parameters are extracted for each video [2].

The video shot segmentation system uses mathematical characterization of cuts and dissolves in the video [1].Different kinds of transitions may occur. An abrupt transition is found in a couple of frames, when stopping and restarting the video camera. A gradual transition is obtained based on effects, such as fade in i.e. a gradual increase (decrease) in brightness or dissolves i.e. a gradual super-imposition of two consecutive shots. Abrupt transitions are obtained for two uncorrelated successive frames. In gradual transitions, the difference between consecutive frames is reduced. Considerable work has been reported on the detection of abrupt transitions

A method for key frame extraction [6] which dynamically decides the number of key frames depending on the complexity of video shots and requires less computation. Priya and Shanmugam describe a method for feature extraction which provides the steps for extracting low level features [4]. The spatial distribution of edges is captured by the edge histogram with the help of sobel operators. Color histogram is the most extensively used method because of its usage in various fields. The color histogram value are recognized using hsv color space. Texture analysis algorithms are used in random

field models to multi resolution filtering techniques such as the wavelet transform. Several factors influence the utilization of Gabor filters for extracting textured image features. The feature library stores the extracted features.

3. System Overview

The architecture of our proposed system is shown in Fig 1. In the offline process, set of videos are given as input and features are extracted from the video**.** In the online process, the features are extracted from the query clip and matched against the feature vectors stored.

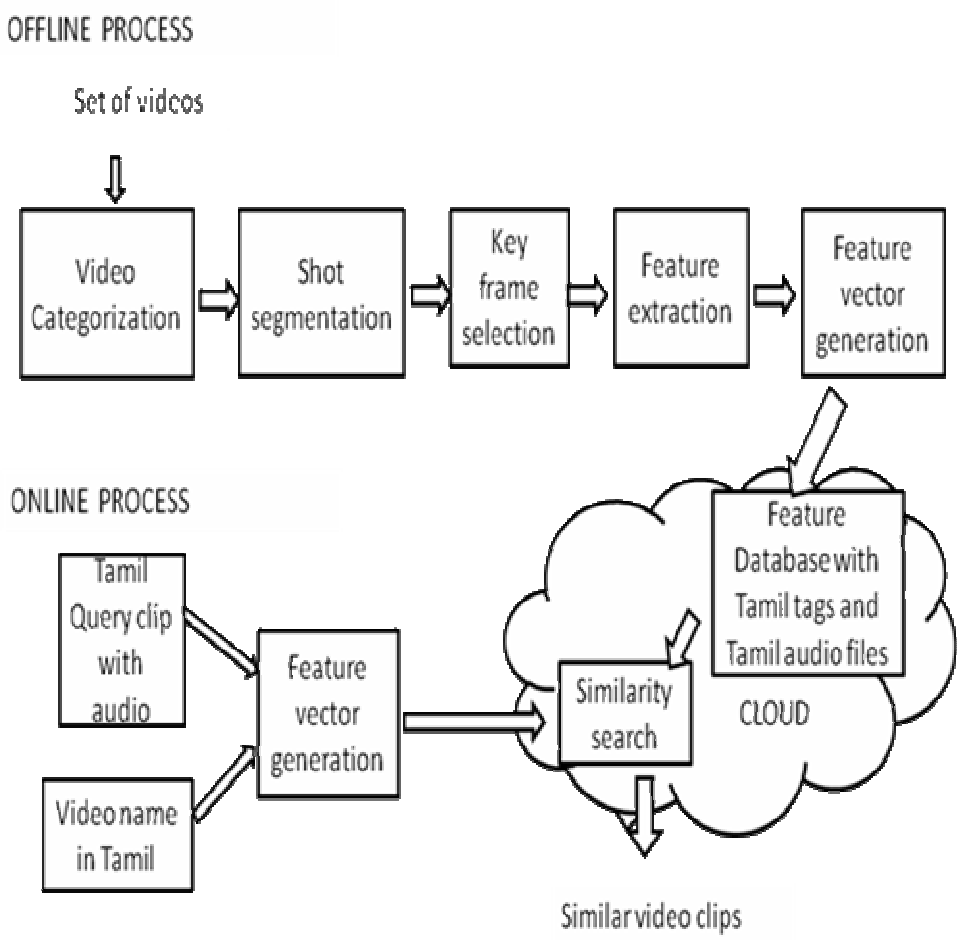


Fig 1 System Architecture

A. Video Categorization

The first process to be carried out is video categorization which is shown in Fig 2. The content based video categorizing method uses camera motion parameters. This parameter helps to categorize the sports videos for identifying different sports types. Camera motion parameters are changing the state among 2 types (Fix and Pan) along with time scale in video sequence. Here, motion vector are classified and histogram is calculated. By analyzing the characteristics of this histogram, camera motion parameters are extracted for each MPEG video.

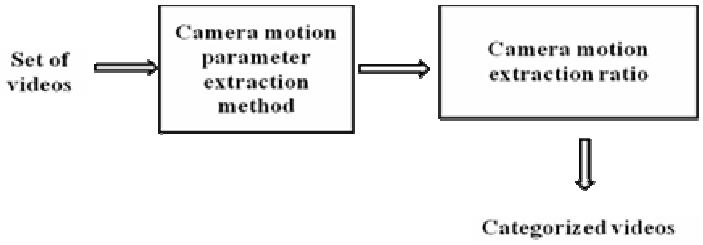


Fig 2 Video Categorization Process

In a video, panning is the sweeping movement of a camera across a scene and Fix means the static position of the camera. For this parameter, camera motion extraction ratio is calculated.

camera motion extraction ratio w[x]

$w[x] = (\text{Num.appear} / \text{Num.total}) * 100\%$

$x = \{FIX, PAN\}$

where,

Num.appear -> number of times of an appearance for camera work x.

Num.total -> total number of frames in the given video.

B. Shot Segmentation

To segment the shots, the video has to be split into video shots prior of conducting any video object analysis. Scene change detection, either abrupt scene changes or transitional (e.g. dissolve, fade in/out, wipe) is employed to achieve the video shot separation.

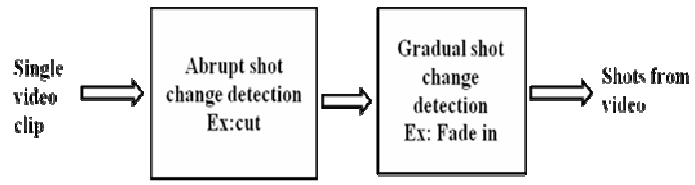


Fig 3 Shot segmentation Process

The proposed algorithm is based on the computation of an arbitrary similarity measure between consecutive frames of a video. The first phase of the algorithm detects the abrupt shot-change detection and second phase detects the gradual transition.

C. Key frame Extraction

A key frame is a frame that represents the content of a shot. This content is the most representative one. In the large amount of video data, first reduce each video to a set of shots and find the representative frames. Each shot obtained by video segmentation algorithm contains a set of frames.

These segments are represented by two dimensional representative images called key frames that greatly reduce amount of data that is searched. Key frames from each shot are obtained by comparing the color information between adjacent frames. A frame will be chosen as key frame if the value exceeds certain threshold.

D. Feature Extraction

Feature extraction is an area of image processing which involves using algorithms to detect and isolate various desired portions of a digitized image or video stream. Different kinds of video features, including edge and color for each key frame is being extracted. To minimize the dimensionality of the data, feature extraction is employed which extracts discriminative features of data.

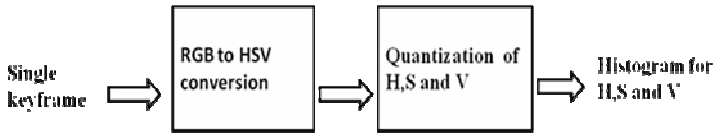


Fig 4 Color Histogram Process

Fig 4 shows the process of color histogram creation. Color histogram is the most extensively used method because of its robustness to changes due to scaling, orientation, perspective, and occlusion of images, which are recognized by using the HSV color space.

Edges in the key frames are detected based on the canny edge detector. The Canny operator works in a multi-stage process. First of all the image is smoothed by Gaussian convolution. Then a simple 2-D first derivative operator is applied to the smoothed image to highlight regions of the image with high first spatial derivatives. Edges give rise to ridges in the gradient magnitude image.

E. Similarity matching

The query video is categorized and key frames are extracted. The color and edge features extracted are matched against the features in the repository. The color features are matched based on the naive similarity algorithm and edge features are matched based on region based histogram.

The algorithm first calculates the color histogram for the query clip and compares with the video set. Each key frame feature vector of query clip is matched with all the feature vectors in the repository and most similar match is retrieved. The histogram values contain mean, entropy and standard deviation of color. From the mostly matched key frames the edge histogram is calculated and matched against query clip. The edge histogram contains region information. The key frames which give the most similarity values are selected and the corresponding videos are retrieved as the similar videos for user query clip.

F. Audio Processing

The next way of Tamil video retrieval focuses on audio processing. The audio track is extracted from the Tamil video as the first step. The audio files are segmented in order to remove the silence and noise. The audio files of each video are processed and the words are extracted and stored as .wav files.

These .wav files are called as features of the audio content.

The user gives the query Tamil video clip as input. This input file contains both audio and video information. The audio data will be segmented to remove silence and extract key words. These key word files are pattern matched against all the .wav files in the feature set. The most matched patterns are found and the corresponding videos are extracted.

The pattern matching of wav files are performed and the results which exceed certain threshold are taken as the result.

F. Text Processing

The next way of video retrieval is based on Tamil text. The wav files of audio input are chosen and are tagged with Tamil text. The user input of Tamil text is transliterated and is searched against the feature set. The matched results corresponding video are retrieved and given as result to user. Transliteration is the practice of converting a text from one language into another language phonetically. Transliteration is different than translation. The Table 1 shows some transliterated English word for tamil word.

Tamil word	Transliterated English word
கடினம்	Kadinam
பூக்கள்	Pookkal
குழந்தை	Kuzhandhai
பாப்பா	Paappa
மழை	mazhai

Table 1: Transliteration of Tamil to English

F. Cloud setup

Eucalyptus is an open source cloud computing system.

The eucalyptus open source cloud environment is setup in Linux cluster.The eucalyptus software is installed.The cloud controller,cluster controller,walrus and storage controller are installed.

The cloud controller is the entry point into the cloud for users and administrators.It asks node managers for information about resources,makes scheduling decisions and implements them after requesting to cluster controller.

The cluster controller executes on a cluster front end machine,or any machine that can communicate to both the nodes running Node controllers and to the machine running cloud controller.Cluster

controllers gather information about and schedules virtual machine execution on specific node controller and also manages virtual instance network.

The Node controller is executed on every node that is used for hosting virtual machine instances.They control the execution,deplyment and termination of virtual machine instances on the host where it runs.

The storage controller is capable of interfacing with various storage systems.It is a block device and it is attached to an instance file system.Walrus allows user to persistent data,organized as buckets and objects.It provides a unique mechanism for storing and accessing virtual machine images and user data.

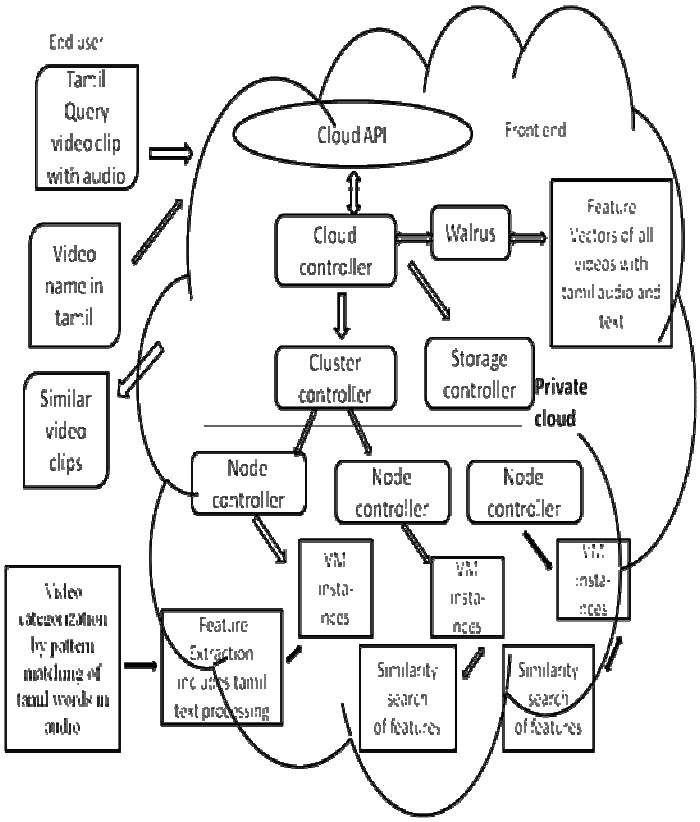


Fig 5 Video Retrieval process in Eucalyptus cloud

The Tamil video retrieval process is developed as application and this application is bundled to the virtual machine instance. The application bundled virtual machine image is uploaded and registered to the eucalyptus cloud. The instances are communicated and the application is run over the cloud. The query video clip is given as input in the cloud front end. The videos are categorized, the key frames are extracted and the similarity search is performed in separate parallel instances. The retrieved video result are given as output to the user.

4. Simulation and Results

The video retrieval process includes video categorization, key frame extraction, feature extraction and similarity matching. The process is carried out in Java media framework and Java advanced imaging.

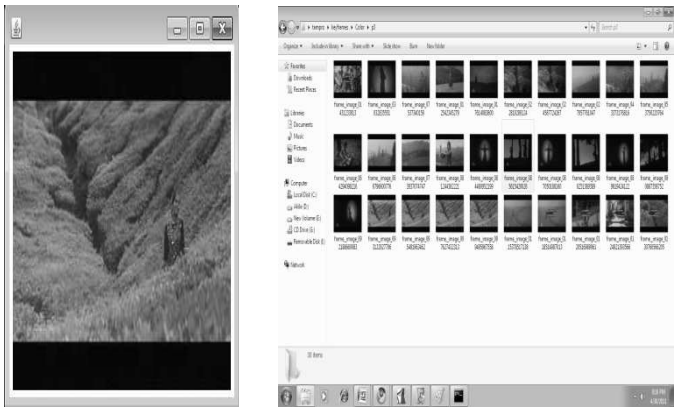


Fig 6 Tamil Quer Video and Key frame extracted from videos

Fig 6 shows the key frame extracted for a given tamil video and Fig 7 shows the categorization and similar video result

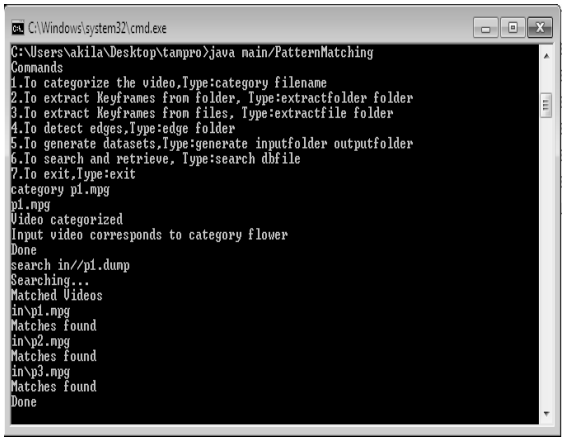


Fig 7 Similarity result of query video

The user gives the query video name as input and based on the commands the videos will be categorized, extracts key frames and features. The similar video will be retrieved if they give search command

5. Performance Evaluation

The video retrieval process is performed in cloud and the performance is evaluated while running in two instance.

No. of videos in dataset	Execution time in 2 instances	Execution time in 1 instance
10	5206.2	18369.69
15	5522.63	18924.41
20	6202.2	21731.45
25	7291.7	25319.52
30	8575.6	28904.35

Table 2:Relation between execution time in one instance and two instance

The application is run in EUCALYPTUS private cloud and the execution time is calculated while running in single instance and two instances. The execution time is much less when we run in two instances. This shows that the video retrieval process shows better performance in cloud.

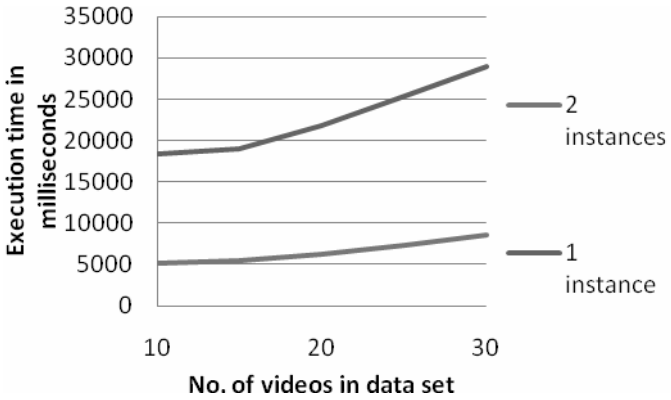


Fig 8 Performance graph in cloud environment

The performance of video retrieval process is checked by precision recall graph.

Recall = DC/DB and Precision = DC/DT

Where DC is the number of similar clips which are detected correctly, DB is the number of similar clips in the database and DT is the total number of detected clips.

Query video	Recall	Precision
Q1	0.1	0.9
Q2	0.35	0.78
Q3	0.39	0.69
Q4	0.6	0.4
Q5	0.8	0.2

Table 3:Precision and recall for query video clips

The precision and recall for various query video clips are computed. The efficiency of the video retrieval process is improved as the retrieval process includes categorization process.

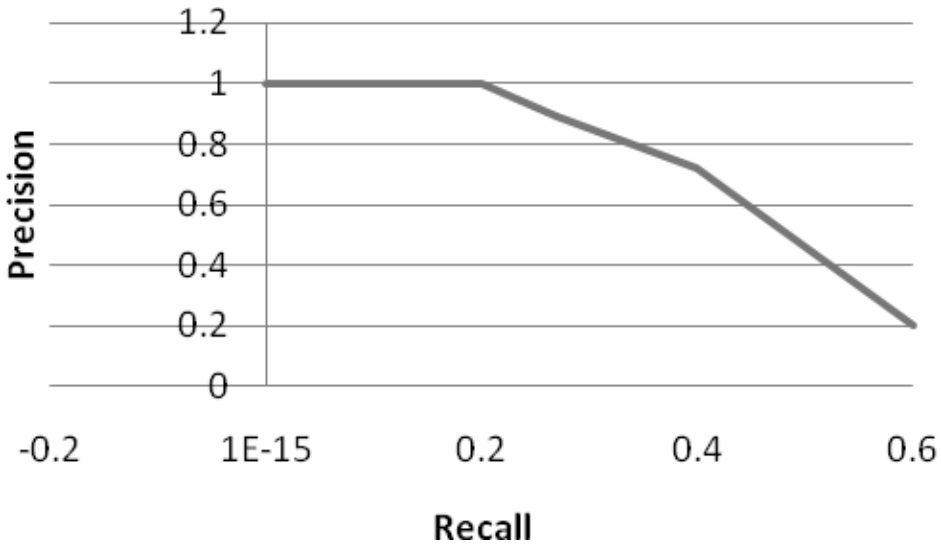


Fig 8 Performance graph for video retrieval

The performance of Tamil video retrieval shows that the most similar videos are retrieved. Also the application in cloud environment shows that the cloud computing provides better performance through execution time and resource sharing.

6. Conclusion and Future work

The proposed video retrieval categorizes the video into different category based on camera motion parameters. It facilitates the segmentation of the elementary shots in the video sequence proficiently. Then the key frames are extracted from the video shots. Subsequently, the extraction of features like edge and color histogram of the video sequence is performed and the feature library is employed for storage purposes.

Then Video retrieval system based on query video clip is incorporated within the cloud. Cloud computing, due to its high performance and flexibility, is under high attention around the industry and research and reduces the computation complexity of Video retrieval process based on visual, audio and text input.

References

- Albanse M., Chianese A., Moscato V. and Sansone L., "*A Formal Model for Video Shot Segmentation and its Application via Animate Vision*",In Proceedings of Multimedia Tools and Applications, Vol 24, 2004, pp. 253–272.
- Dobashi K., Kodate A. and Tominaga H., "*Camera Working Parameter Extraction for Constructing Video Considering Camera Shake*",In Proceedings of International Conference on Image Processing (ICIP), Vol.III, 2001, pp.382-385.

- Nurmi D., Zagorodno D., Youseff L. and Soman S., " *The Eucalyptus Open source Cloud-computing System"*, In Proceedings of International Symposium on Cluster Computing and the Grid,2009.
- Priya R. and Shanmugam T.N.,"*Enhanced content-based video retrieval system based on query clip*", In Proceedings of International Journal of Research and Reviews in Applied Sciences ,Vol 1, 2009.
- Takagi S., Hattori S., Yokoyama K., Kodate A. and Tominaga H.,"*Sports video categorizing method using camera motion parameters*", In Proceedings of Visual communications and Image processing, Vol 5150, 2003, pp.2082-2088.
- Zeng X., WeimingHu , Wanqing Li,Zhang X. and Xu B., " *Key frame extraction using dominant set clustering*", In Proceedings of International conference on Multimedia & Expo(ICME), 2008.

Animated Story Visualizer for Tamil Text

M. Janani, Dr. T. Mala
Department of Information Science and Technology
Anna University, Chennai, India
janani_mrl@yahoo.com, malanehru@annauniv.edu

Abstract

Natural language is a straightforward and efficient medium for describing visual facts and mental images. The System uses a novel approach to generate animation from Tamil texts such as stories. Tamil text is pre-processed and the necessary features like named entities, environmental constraints, temporal and emotion constraints for the given stories are extracted and placed in the database. The system automatically generates a query based on the users input and compare it with features stored in the database. Finally animation is dynamically generated using an external motion synthesis system. Using this system, even greenhorn users can generate animation quickly and easily by giving the Tamil text.

***Keywords*— Computer animation, Natural Language Processing, Pre-processing, Feature Extraction, Motion synthesis**

I. INTRODUCTION

These days, animations are widely used in many applications, such as cartoons, web graphics, games, and so on. Computer animation is one of the best methods for depicting the dynamic content. A medium is necessary for the animation to be created in a convenient and natural manner. It should be possible to describe the scenes directly from natural language.NLP is an easy and effective way to analyze, understand and generate languages that humans use naturally.

The aim of this work is to generate an animation from Tamil texts such as movie scripts or stories. Training input text is given to the pre-processing module. Here tokenization is performed and the tokens are given to the morphological analyser which is used to convert the tokens into a POS tags. Information related to named entities, temporal constraints, emotion and environment inference features are extracted. A query is generated automatically from the input text which contains information for the search process and compares it with the information already stored in the database. Finally motion synthesis generates an animation. The interactions between characters are handled by this module based on the information provided in the database.

II. RELATED WORK

Generating animation from natural language texts has been a challenge. WordsEye developed by Coyne and Sproat [1] converts natural language texts to a scene. WordsEye focuses on generating a still image, when a character motion is specified in a given text, the system simply prefer to pose for

the action generated from the database. The Carsim system [2] describes a new version of text-to-scene converter that handles texts describing car accidents using computer program and it is visualized in the 3D environment. Storytelling System [3] illustrates a system called Interactive e-Hon, which provides storytelling in the form of animation and conversation translated from original text. A Constraint based scene conversion system [4] describes a Text2Scene conversion method which automatically converts text into 3D scenes. A large database of 3D models is used by this method to depict entities and actions.

III. SYSTEM OVERVIEW

In this section, overview of our system (Figure 1) is given, where the major components are identified. When the Tamil text is given to the system, Tamil text is pre-processed and the information are extracted and stored in the database along with the objects created. When an input text is given to the system it automatically generates the query from the input text and compares it with the information stored in the database. An animation is then generated using an external motion synthesis system.

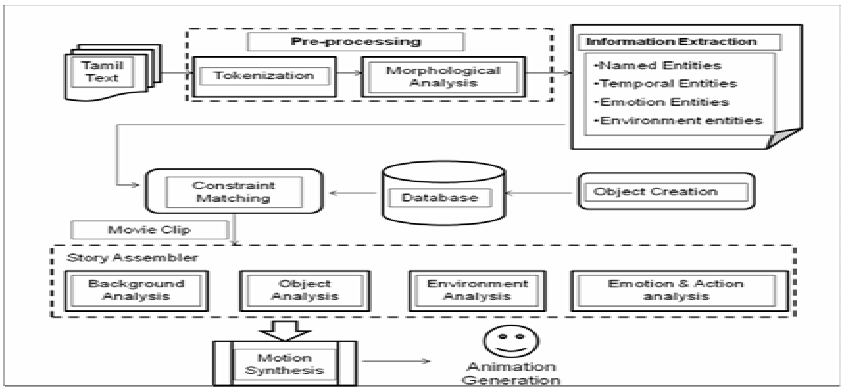


Figure 1. SYSTEM OVERVIEW

PRE-PROCESSING

When the Tamil text is given to the system, natural language processes (Tokenization and morphological analysis) are applied first.

Tokenization

The first step in NLP is to identify tokens, which decomposes the delimiters like punctuation and whitespaces. Here Tamil text is given as the input to the tokenizer which breaks the text into meaningful tokens. The tokens generated by the tokenizer are passed to the analysis engine.

Morphological Analysis

RCILTS [5] developed a tool called Atcharam, an analyser which performs Morphological Analysis for Tamil text. The Morphological analyser takes a derived word as input and separates it into root word and associated morphemes. It is the basic tool used in spell checker, grammar checker, parser and machine translation systems. It has two major modules noun analyzer and verb analyzer.

வண்டுகளுக்கு	வண்டு< noun >
	கள்< plural >
	உக்கு<case marker >

படித்தான்	படி< verb >
	த்த்< past tense marker>
	ன்< gender >

Tamil Noun and Verb classification example

By this method morphemes are generated and given to the learning process where the necessary informations are extracted.

INFORMATION EXTRACTION

OBJECT IDENTIFIERS

Object Identifiers recognize named entities in text by Named Entity Recognition (NER). "Rule based approach" is used to extract named entities from the given text. Initially, root words say Noun, verb, adjective, pronoun, adverb from text file are extracted. Rules are created based on prefix and postfix of noun, i.e. noun that occurs between verb and noun, noun that occurs between noun and noun, noun that occurs between noun and verb and so on. If any of the above rules satisfies the input text named entities are extracted. Here is an example,

Input: ஒரு குளத்தில் எறும்பு தத்தளித்து

Given input text is pre-processed and the root words are extracted.

குளம்<Noun>
எறும்பு<Noun>
தத்தளி<Verb>

Now the rules are applied to this extracted root words. Here எறும்பு comes between noun and verb which satisfies the rule is extracted.

TEMPORAL AND EMOTION EXTRACTION

Temporal reasoning in NLP involves extraction, representation and reasoning with time and events in the natural language text. Here to extract temporal constraints, "manually created dictionary" is used. The root words are compared with the manually created dictionary and temporal constraints are extracted if the input text satisfies the inferences present in the dictionary. Similarly different emotion present in the text is also extracted using manually created dictionary.

Figure 2 shows the different emotional constraints to be depicted.

HAPPY	இன்பம்,மகிழ்ச்சி,குதூகலம்,சந்தோஷம்,ஆனந்தம்,களிப்பு,பெருமிதம்,சிரிப்பு
ANGER	அகங்காரம்,சினம்,கோபம்,வெறுப்பு,எரிச்சல்,சீற்றம் தாபம்
SURPRISE	அதிசயம்,ஆச்சரியம், பிரமிப்பு,மலைப்பு, வியப்பு
FEAR	அச்சம்,பயம்,பீர்,பொருமல்,விதிர்ப்பு,கவலை,அஞ்சு,கலக்கம்,நடுக்கம்
SADNESS	சோகம்,அழுகை,சோர்ந்த, துயரம்,வருத்தம், துன்பம், கண்ணீர், கூச்சல், அலறு, கதறு,புலம்பு,கத்து, முழக்கம்,கூக்குரல், துக்கம், வாட்டம், விசனம்

Figure 2 Emotion Constraints

Finally, environmental constraints that specify location and actions are extracted.

ANIMATION GENERATION

Movie clips for the extracted information are created using Adobe Flash professional and stored onto the database.

CONSTRAINT MATCHING

When an input text is given, information constraint should match with the information present in the database to generate animation. String matching algorithm is used to compare the information extracted and information stored in the database. Let P[1...M] and T[1...N] be the character array for the given string. Pattern *P* is said to occur with shift *s* in text *T.* To find all valid shifts or possible values of s so that *P[1..m]* = *T[s+1..s+m]* ; There are *n-m+1* possible values of *s*.

Procedure String Matcher(T,P)

1. $n \leftarrow$ length [*T*];
2. $m \leftarrow$ length[*P*];
3. for $s \leftarrow 0$ *to* $n-m$
4. do if *P[1..m]* = *T[s+1..s+m]*
5. then shift *s* is valid

Find first match of a pattern of length M in a text stream of length N.

The extraction of Pattern கா க ம் is done by,

காகம் M = 4

க ழு தை ஆ டு கா க ம் கு ர ங் கு

கா க ம்

கா க ம்

கா க ம்

கா க ம்

கா க ம்

கா க ம்

By this method exact string is matched from the database for the given information.

STORY ASSEMBLER

Storyboards are the only way to convey rich information, viewing a particular order of events in a most appealing way. Basically the system searches for noun and verb from the given input text then automatically assemble and analyse the subsequence like background, named entities, temporal, emotion and action movie clip from a database that matches the constraints.

MOTION SYNTHESIS

Animation is generated by motion synthesis by efficiently connecting the movie clips that are assembled by the story assembler from the database. The character and objects interactions are handled by this module based on the information that the movie clips have. Timeline specifies what

kind of action occurs at particular time. Once the timeline has been set, animation is generated for the given Tamil text. Figure 3 shows the animation generated for the given Tamil Text

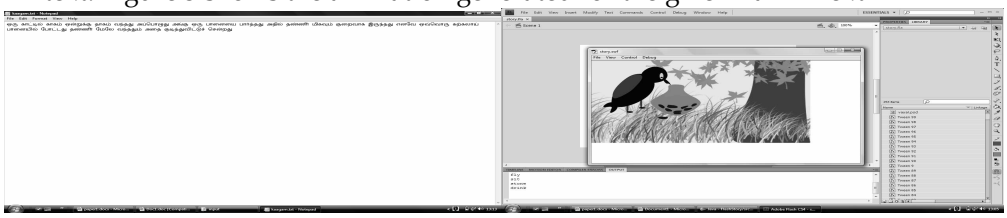


Figure 3 Animation is generated from Tamil text

PERFORMANCE ANALYSIS

The performance analysis is used to monitor the functioning, efficiency, accuracy and other such aspects of a system. For the analysis performed for the learning process, the overall accuracy obtained is 83%. Figure 4 shows the Performance analysis for Animation generated from the Tamil text.

	No.of Stories	Exactly generated	Approx. generated
viewer1	18	15	3
viewer2	18	13	5
viewer3	18	14	4
viewer4	18	16	2
viewer5	18	14	4

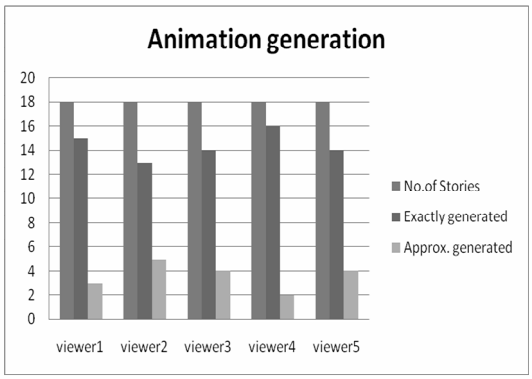


Table 1 Test case for Animation Generation *Figure 4 Performance analysis for Animation generation*

The overall accuracy obtained for the generation of animation is 80%. The performance can be further improved by generating rules and optimizing the learning process.

CONCLUSION AND FUTURE WORK

The system provides automated generation of animation from Tamil text which provides a new approach for users to create animation quickly. The proposed method takes Tamil text as the input and it is pre-processed and features like named entities, temporal constraints, emotion and environmental constraints are extracted and animation is generated dynamically by motion synthesis. Even non-professional people can rely on this system and they can generate animation quickly and easily by giving the Tamil text. Future work can be extended by generating animation via automatic speech recognition rather than text.

REFERENCES

- Coyne.B and Sproat.R, "*Wordseye: an automatic text-to-scene conversion system*", In Proceedings of SIGGRAPH 2001, pp. 487-496.
- Johansson.R, Berglund.A, Danielsson.M and Nugues.N, "*Automatic Text-to-Scene Conversion in the Traffic Accident Domain*", In Proceedings of The Nineteenth International Joint Conference on Artificial Intelligence, 2005, pp. 1073–1078.
- Kaoru Sumi, Mizue Nagata, "*Animated Storytelling System via Text*"**,** In Proceedings of SIGCHI International Conference on Advances in computer entertainment technology,2006.
- Liuzhou Wu and Zelin Chen,"*A Constraint-based Text-to-Scene Conversion System*", In Proceedings of International Conference on Computational Intelligence and Software Engineering 2009.
- Anandan, P., Ranjani Parthasarathy & Geetha, T.V., "*Morphological Analyser for Tamil*", ICON 2002, RCILTS-Tamil, Anna University, India.

Popularity Based Scoring Model for Tamil Word Games

Elanchezhiyan.K, Karthikeyan.S, T V Geetha,
Ranjani Parthasarathi and Madhan Karky
chezhiyank@gmail.com, tv_g@hotmail.com, rp@annauniv.edu, madhankarky@gmail.com
Tamil Computing Lab (TaCoLa),
College of Engineering Guindy, Anna University, Chennai.

Abstract

In this paper we propose a Popularity based Scoring model for Tamil word games. Games are one of the effective means to teach a language. There exist very few online word games for Tamil. Scoring is one of the key aspects of a game that nurtures interest in the player apart from the interface and logic. The Popularity Based Scoring Model proposed in this paper, uses a word statistics crawler to periodically collect the statistics of word usage in popular blogs, news articles and social nets. The popularity of every word is thus modeled in comparison with every other word in the language. The model was successfully implemented in a simple unscramble game titled 'Miruginajambo'. Over three lac root words from Agaraadhi Project were crawled for statistics and 20000 words were obtained for the game based on threshold levels for increasing levels of complexity in the game. The paper concludes providing the results and analysis of implementing the model and also discusses various word games where this model can be used.

1. Introduction

Word based games can serve as an effective tool to teach vocabulary in any language. The complexity levels, the score achieved motivates a player to play more and there by learn more words. One key challenge in designing such games is the scoring model. A scoring model for a game means a lot more than just a value associated with the game level or complexity. An effective scoring model has to motivate a player to play more and there by retain the player's interest to come back again.

Tamil language has very few online games available online. These games are mostly flashcard-based games. With new words being introduced in various domains such as medicine, computer science and other disciplines, such games can be the most effective way to teach words. The main reason for popularity of word games in English is their scoring models apart from the user interface they build around their games.

In this paper we propose a popularity-based scoring model for word-based games in Tamil. Providing this scoring model we test the scoring model over two games namely 'miruginajambo', a unscramble game and 'thookkuthookki', a Tamil equivalent of hangman. Popularity Based Scoring Model generates score based on the combination of the word's popularity, length of the word and Tamil alphabet popularity.

This paper is organized into four major sections. The following section gives the background about scoring models and other relevant literature. The third section gives the popularity based scoring model and the components of the scoring model. The final section concludes discussing the results.

2. Background

A scoring model calculates scores based on performance of any system on various domains like Credit Scoring Model in baking application, Fuzzy Logic Approaches in game etc. In the new generation mobile multiplayer games, scoring was generated by using Fuzzy Logic Approach [1]. Automatic Target-scoring System of Shooting Game Based on Computer Vision [2], Online Score System Using Hierarchical Colored Petri Nets [3] to evaluate the outer and inner performances of the system, such as scan, score, and resource utilization. The Credit scoring models [4] are developed to classify the loan applicants as accepted or rejected. The decision is based on the information of each applicant such as age, income and debit ratio. First time we proposed The Popularity Based Scoring Model proposed for Word games, uses a word statistics crawler to periodically collect the statistics of word usage in popular blogs, news articles and social nets. This word Popularity was implemented in the Agaraadhi Online Dictionary [5]. The Word from agaraadhi is given to web and found the frequency distribution of the word across the popular blogs, news articles, social nets etc. The Scoring Model uses the Frequency Analysis of Tamil Alphabet [6]. The Frequency Analysis is done by splitting the Tamil word into alphabet, splited alphabet are added to their corresponding counter, frequency of each alphabet was identified individually.

3. A Game Framework based on Popularity-Based Scoring Model

We propose a simple game framework for an unscramble game in this paper, depicted in figure1. The framework can be basically divided into two major divisions, online and offline, in terms of the time of processing. This section describes the various scoring generator in detail.

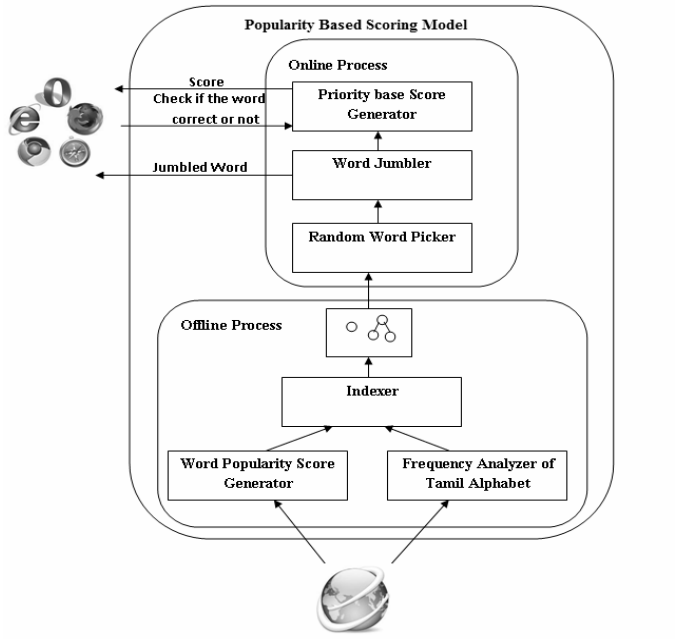


Fig 1: Popularity Based Scoring Model

3.1 Offline Processing

The Offline Process takes a Set of Tamil words as input, Word Popularity Scoring, and Tamil Alphabet Frequency Analysis are the key tasks here.

3.1.1 Bag of Tamil Words

Tamil words are obtained for analysis from the Agaraadhi project comprising over 3 lac words on various domains such as General, Engineering Technology, Literature, Medicine and Computer Science.

3.1.2 Word Popularity Scoring

Word Popularity shows the word usage in the web. The words from agaraadhi are crawled over popular news sites; blog articles and social networking sites periodically and the frequency distribution of the word across sites are identified and recorded. This overall information is then used to compute the popularity score for each word.

3.1.3 Tamil Alphabet Frequency Analysis

The words obtained from the set of words are split into alphabet that constitute the word and the split alphabets are added to their corresponding counter, frequency of each alphabet is identified individually. Alphabet Frequency Analysis generates score based on the frequency value of the alphabets. This Analysis result will be used later to find the popularity of a letter and thus the complexity of a word. So, a low frequency alphabet contained word gets a higher score.

3.2 Online Processing

The Online Process Comprises of Random word Picker, Word Jumbler and Popularity Based Score Generator

3.2.1 Random word Picker

Random word Picker fetches a random word based on the domain specified by the user and the Word Popularity. It will be possible now to decrease the word popularity score on every higher level to increase the complexity of the game.

3.2.2 Word Scrambler

Word Scrambler module scrambles the Tamil alphabets in a word such that all alphabets are not placed in their actual correct spots. The jumbler is randomized such that the next time the same word will be jumbled in another combination.

3.2.3 Popularity Based Score Generator

Score Generator generates the score based on the proposed Popularity Based Scoring Model using the parameters such as word popularity, level of the game, Low frequency scored occurrences of Tamil Alphabets in a word, total number of swaps to complete a level and time taken to complete a level. Let w be the word, l_w be the length, Let P_w be the popularity score of the word, Let P_a be the average alphabet popularity frequency of the alphabets in w, Let t denote the time taken to solve the word in

seconds, Let s denotes the number of swaps needed to solve the word, Let α be the score for a perfect answer.

The score for a solving,

$$\text{Score } S(w) = \alpha * (1-P_w) * (1- P_a) * \left(1 - \frac{t}{60}\right) * \left(1 - \frac{s}{lw}\right)$$

4. Results and Discussion

The framework depicted in figure 1 was implemented with a simple web interface. The snapshots of the working game with the popularity based scoring model are given in figure 2. The same game was developed with and without the popularity based scoring model with the earlier version giving a score just based on the level and number of letters swapped. The version with popularity based scoring model was found to receive more users playing for a longer time and repeatedly as they find their current score rapidly increase if they identify a less popular word.

5. Conclusion

In this paper we proposed a Popularity-Based Scoring Model for computer based word games in Tamil. The popularity of words was identified by their usage over the internet by news, blog and micro blog writers. This information is then converted to scoring every word in a dictionary. This score is used in the Scoring model to compute the score for the user at different levels. The scoring model is compared to a traditional level based scoring model. Analyzing user behavior over the two models we conclude that the popularity based scoring model creates more interest in user to play the game for long time and repeatedly compared to the traditional level based scoring.

Fig 2: Snapshots of the working game with the popularity based scoring model

References

- Alan Graf, Siemens d.d, Fuzzy Logic Approach for Modelling Multiplayer Game Scoring System, ConTEL 2005.
- Design of Automatic Target-Scoring System of Shooting Game Based On Computer Vision, Xinnan Fan, Qianqian Cheng, Changzhou, Jiangsu Province, IEEE International Conference on Automation and Logistics Shenyang, China August 2009.
- Yang Xu , Xiaoyao Xie, Daoxun Xia1, Zhijie Liu, Lingmin Chen1, Modeling and Analysis of an Online Score System Using Colored Petri Nets, Anti-counterfeiting, Security, and Identification in Communication, 2009. ASID 2009. 3rd International Conference, Aug. 2009.
- Entropy multi-hyperplane credit scoring model, Wasakorn Laesanklang, Krung Sinapiromsaran Boonyarit Intiyot, International Conference on Financial Theory and Engineering, 2010.
- Agaraadhi – An Online Tamil-English Dictionary http://www.agaraadhi.com, Last Accessed Date on 28th April 2011.
- Karthika Ranganathan, Elanchezhiyan.K, T.V Geetha, Ranjani Parthasarathi & Madhan Karky, The Frequency Analysis of Tamil Alphabet, National Seminar on Computational Linguistics and Language Technology, March, 2011.

Multilingual Cross – Domain Classification of Tamil Web Documents based on Neural Network with Dimension Reduction

M.Balaji Prasath[1], Dr.D.Manjula[2]
itprasath@gmail.com[1], manju@annauniv.edu[2]
Department of Computer Science and Engineering,
Anna University, Chennai.

Abstract

Automatic classification of web document increases in the regional languages, because of amount of information available in the regional languages (like Tamil, Telugu, Hindi) is huge in the internet in the form of e-Book, news, articles and other type of formats. It is difficult to categorize those documents based on the subject of interest. Tamil is a Rich Dravidian language, it have a millions of documents in the Web Repository, due to growth of digital documents, categorization needed to classify document. Too much classification techniques are present for the English documents classification like SVM, K-NN, Decision trees, Neural Network technique, but classification in regional languages like Tamil, it's new and emerging. So that our proposed work involves first, genetic algorithm will be employed to reduce dimension of document .Second, Multilingual Cross- domain classification, involves the predefined labels in the English language will be used to classify the Tamil Corpus, because pre-defined labels in the source domain is expensive to create, so that look for other domain of same interest to classify the documents. Third Back Propagation Technique applied to classify Corpus.

Key Terms: Classification, Multilingual, Cross-Domain, Dimension Reduction

Introduction

Today most of the documents exist in the electronic repository like e-books, journals, news articles and other sources of information in form of English only. This electronic document exits in other regional languages also (like Tamil). To classify those Region documents lot of research going on.

Tamil[1] is an oldest regional language present in the world. Around billion of people speaking Tamil and lot of documents present in the Tamil language. Natural Language processing of Tamil is difficult, because of little bit research is taken place. To analyze their keywords, linguistics plays an important role. Lot of research already taken place to classify English documents based on supervised and unsupervised learning. I.e. two types learning is their (i) supervised leaning means of classification documents based on the pre-defined label categorization. It first train the training document based upon the pre-defined labels, and test the test documents and classify based upon the training set. (ii) Unsupervised learning is a clustering.

Many machine learning technique available like SVM, KNN Classifier, Neural Network, Bayesian Classifier based on mathematical approaches. For Pre-label is expensive, to avoid that other domain

label is used for classification purposes, but in English document collection lot of Cross-Domain[10] Labels are available, in order to classify the documents in other domain but in Classification based on rare. So that use labels in the domain of English Document, to the corresponding labels in Tamil documents, it reduce the classification effort, and also produce better results.

Before using those approaches dimension reduction plays an important role to minimize the no of keywords present in the document. Our proposed approach use the genetic algorithm for reduction of no of key attributes present in the documents.

Dimension reduction carried out based on feature selection and feature reduction methods.

Feature selection [2] means that, it selects the keywords based on attributes, which contribute reduction of no of words in the documents. Selection plays an important role here. Two types present (i) filter method Separating the feature selection from the classifier learning, and relay on general characteristics of data, no bias over any learning algorithm, generally it fast. (ii) Wrapper model, relaying on predefined classification algorithm, and computationally expensive.

Genetic algorithm [3] will be used as a dimension reduction technique, it takes the set of keywords as a population of terms, and neural network will be employed as a classifier, which train and classify the training documents and classify testing documents after that training phase.

Tamil corpus will be generated automatically, by using a web crawler. Crawler return the set of document pages (Tamil) particularly news articles. These collectively articles used to form the corpus. Further classification will be done using those Tamil news articles (Corpus).

This paper section 2 describes the web crawler section 3 describes the Tamil corpus, section 4. describes the dimension reduction using the genetic algorithm, section 5, describes the classification using neural network.

2. Web Crawler

Crawler is a software program, which can fetch the WebPages based on the seed URL given to the Crawler. Here seed URL will be "Tamil news article" site URL. This crawler crawls only site given to the input to the crawler, it doesn't navigate to other site. It uses a muti threaded downloader to down load the web pages , based on that seed URL given to the crawler, this crawler, crawls the pages only within that link. Suppose it should news.goole.com means, it crawl the link fully, retrieve the document within that.

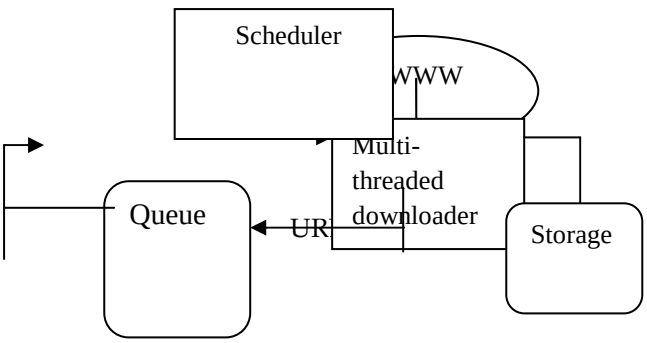


Figure 1. Architecture of Web Crawler

3. Tamil Corpus

Many research going on to build a corpus in Tamil. Central institute of Indian language (CIIL)[4], Mysore actively involved in building the corpus in different regional languages.

Here, corpus build by using a web crawler, it crawl a web pages and stored it in a local database. After that it will be edited in order to make and formed as a corpus

Tamil has 12 vowels and 18 consonants. This are combined with together 217 composite characters and 1 special characters counting to the total of 247 characters. To build a corpus for that rich type of grammar is too difficult. So that crawler will used to retrieve web content, and it edited to form a corpus.

4. Dimension Reduction using genetic algorithm

Generally dimension reduction used to reduce the no of words in a corpus. Because corpus have a huge collection of words, but few collections of words in a corpus makes the document meaningful, So that to identify those words, dimension reduction plays a vital role.

Genetic algorithm used as an optimization technique. Here it plays as a dimension reduction, it choose a set of attributes like content name, sub-content title, and other.

Genetic algorithm uses a input as a set of population of attributes, instead of choosing a single attributes, it reduce the no of words from a thousand to hundred, each attribute like a gene, group of attribute forms a chromosome, uses a various operation like,

1. Crossover: single or multi-point
2. Mutation
3. Reproduction

4.1 Algorithm of GA for dimension reduction

Step1: form the set of attribute as a chromosome.

Step2: generate the fitness function for each gene in the population.

Step3: apply the genetic operator, and evaluate the fitness once again.

Step4: stop, if attain the terminating condition, else generate new population and go to step2.

5. Classification of Web Document

After the Identification of set of key attributes, need to classify the training documents using a neural network. Neural network [9] have a set of input nodes, and hidden nodes, and a corresponding output nodes. Training documents taken as a input to the system, that should be trained and classified accordingly based on the attributes generated by the genetic algorithm, theoretically says that, dimension reduction after that classification improve the result future

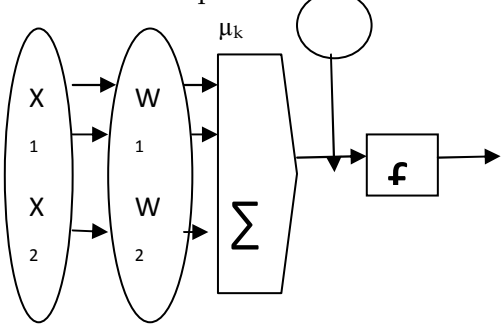


Figure 2. Architecture of Neural Network.

Back propagation technique employed in the classification of web documents. Feedback given to the neural network with every set of documents trained. After that documents tested against the network, whether it should be classified correctly. Theoretical performance is better than other classification technique.

Conclusion and Future Work

Automatic classification of Tamil web content increase the need for separate classification approaches, for that genetic algorithm employed as a dimension reduction technique, and classified accordingly based on the selected attributes, it improve the precision and recall after the dimension reduction.

Future improvement in the neural network, will be use of winnow/preceptor technique with no hidden layer improve classification technique.

References

- K. Rajan, V. Ramalingam, M. Ganesan, S. Palanivel, B. Palaniappan, Automatic classification of Tamil documents using vector space model and artificial neural network, Expert Systems with Applications 36 (2009) 10914–10918.
- Nan Du, Hong Peng, Wenfeng Zhang,Application of Modified Genetic Algorithm in Feature extraction of the Unstructured Data, International Conference on Advanced Computer Control,IEEE 124-128.
- Philomina Simon, S. Siva Sathya, Genetic Algorithm for Information Retrieval
- M. Ganesan,Tamil Corpus Generation and Text Analysis
- M. Selvam, and A. M. Natarajan,Language model adaptation in Tamil language using cross-lingual latent semantic analysis with document aligned corpora, CURRENT SCIENCE, VOL. 98, NO. 7, 10 APRIL 2010
- Thair Nu Phyu, Survey of Classification Techniques in Data Mining, Proceedings of the International MultiConference of Engineers and Computer Scientists 2009 Vol I,IMECS 2009, March 18 - 20, 2009, Hong Kong
- S.Kohilavani, T.Mala and T.V.Geetha,Automatic Tamil Content Generation,IEEE IAMA 2009.
- Chih-Ming Chen, Hahn-Ming Lee, Yu-Jung Chang,Two novel feature selection approaches for web page classification, Expert Systems with Applications 36 (2009) 260–272
- Cheng Hua Li, Soon Choel Park,An efficient document classification model using an improved back propagation neural network and singular value decomposition, Expert Systems with Applications 36 (2009) 3208–3215.
- Sinno Jialin Pan*y*, Xiaochuan Ni*z*, Jian-Tao Sun*z*, Qiang Yang*y* and Zheng Chen,Cross-Domain Sentiment Classification via Spectral Feature Alignment, *WWW 2010*, April 26–30, 2010, Raleigh, North Carolina, USA.

On Emotion Detection from Tamil Text

Giruba Beulah S E, and Madhan Karky V

Tamil Computing Lab (TaCoLa),

College of Engineering, Anna University, Chennai.

(ljcsegb@gmail.com) (madhankarky@gmail.com)

Abstract

Emotion detection from text is pragmatically complicated than such recognition from audio and video, as text has no audio or visual cues. Techniques of emotion identification from text are, by and large, linguistic based, machine learning based or a combination of both. This paper intends to perceive emotions from Tamil news text, in the perspective of a positive profile, through a neural network. The chief inputs to the neural network are outputs from a domain classifier, two schmaltzy analyzers and three affect taggers. To aid precise recognition, tense affect, inanimate/animate case affect and sub-emotion affect dole out as the supplementary inputs. The emotion thus recognized by the generalized neural net is displayed via a two dimensional animated face generator. The performance and evaluation of the neural network are then reported.Index Terms—Emotion detection, Machine learning, Neural network, Tamil news text.

I. Introduction

Emotion detection from text attracts substantial attention these days as this if realized, could result in the realization of a lot of fascinating applications like emotive android assistants, blog emotion animators etc. However, emotion detection from text is not trouble-free, as one cannot, with a single glance get a hold of the emotions of the people about whom the text is based. Further, what appears to be sad for a person may perhaps appear fear for someone. Emotion detection from text is thus influenced by the empathy of the readers. Also, as there may be no background information, to shore up the emotion of a person in text, it is better if emotion detection is based on some model empathy.

The aim of this paper is to detect emotion from Tamil news text by a self learning neural network which takes in linguistic and part of speech emotive features. The emotion is identified by assigning weights for features based on their affective influence.

II. Background

Tamil is a Dravidian language as old as five thousand years and enjoys classical language of the world status along with Hebrew, Greek, Latin, Chinese and Sanskrit. Unlike Sanskrit, Latin and Greek, which are very rarely in use, it is a living language and has fathered many Dravidian languages like Malayalam, Telugu etc. It is morphologically very rich and has a partial free word order. It groups noun and verb modifiers, (adjectives and adverbs) under a single category, *Urichols*. Noun participles are equally affective as the verb participles. Thus, apart from parts of speech, morphological entities like cases and participles are also affective.

Among the methods of emotion detection, [1] observes that the Support Vector Machines using

manual lexicons and Bag Of Words approach perform better. The Support Vector Regression Correlation Ensemble is an album of classifiers, each trained using a feature subset tailored to find a single affect class. However, as support Vector machines suffer from high algorithmic complexity and requirement of quadratic programming and do not efficiently model non-linear problems, Neural networks has been opted as they provide greater accuracy than Support Vector Machines.

Nevertheless, Neural networks sport disadvantages like local minima and over fitting. The former can be handled by adding momentum and the latter is usually solved by early stopping. Since the neural net could memorize all training examples if over the need hidden neurons are present, three promising prototypes are constructed, trained and tested to find the optimal one. To this optimal neural net, the affective feature tags from Tamil news text are given, to recognize the inherent emotion, based on their respective affective strength.

Kao, Leo, Yahng, Hsieh and Soo use a combinatory approach of dependency trees, emotion model ontology and Case based reasoning [2] where cases are manually annotated. Such an approach may work for languages of few cases. However, not all cases in a language may be affect sensitive. Cases in Tamil are eight and only two of these are affect sensitive, namely the instrumental and the accusative case. Thus manually annotating cases in Tamil would normally fail as cases aid in increasing or decreasing the affect of the verb nearby.

The separate sub-class networks [10] for Emotion recognition with context independence in speech seem to be promising but for the low precision of 50 even when the learning is supervised using a simple backpropogation network. However, this project is similar to the above in one aspect; in assuming model empathy to support context independence, thereby avoiding the bias on a particular individual involved in the text.

Sugimoto and Masahide [9] split the text into discourse units and then into sentences identifying the emotion of discourse and sentences separately. The language of the text considered is Chinese which is monosyllabic partially with nouns, verbs and adjectives being largely disyllabic. Tamil is partial free word order and does not have such phonological restrictions. Hence, application of such an approach to Tamil may not be fitting intuitively.

Soo Seoul, Joo Kim and Woo Kim propose to use keyword based model for emotion recognition when keywords are present and to use Knowledge based ANN when text lacked emotional keywords [6]. The KBANN uses horn clauses and example data. However, the accuracy of emotion recognition using KBANN is less ranging from 45%-63% compared to the recognition range of 90% where emotional keyword affect analysis is incorporated.

Most of the research in emotion recognition is directed toward audio and video from which one can infer so many cues even without understanding the narration .For text, the features of the language of text has a major emphasis on emotion recognition. Tamil is a morphologically rich language and hence its affect sensitive features would drastically vary with the usually chosen languages for emotion recognition like Chinese which is character oriented and English which is least partially inflected.

III. An architecture for Emotion Recognition

This section throws light on each module and its functions. Tamil Morphological analyzer, a product of the Tamil Computing lab of Anna University is used as the tool to retrieve part of speech like nouns, verbs, modifiers and cases. The 2D animated face generator is another tool to display the found emotion via a two dimensional face.

Following is the architecture diagram of the Neural net framework.

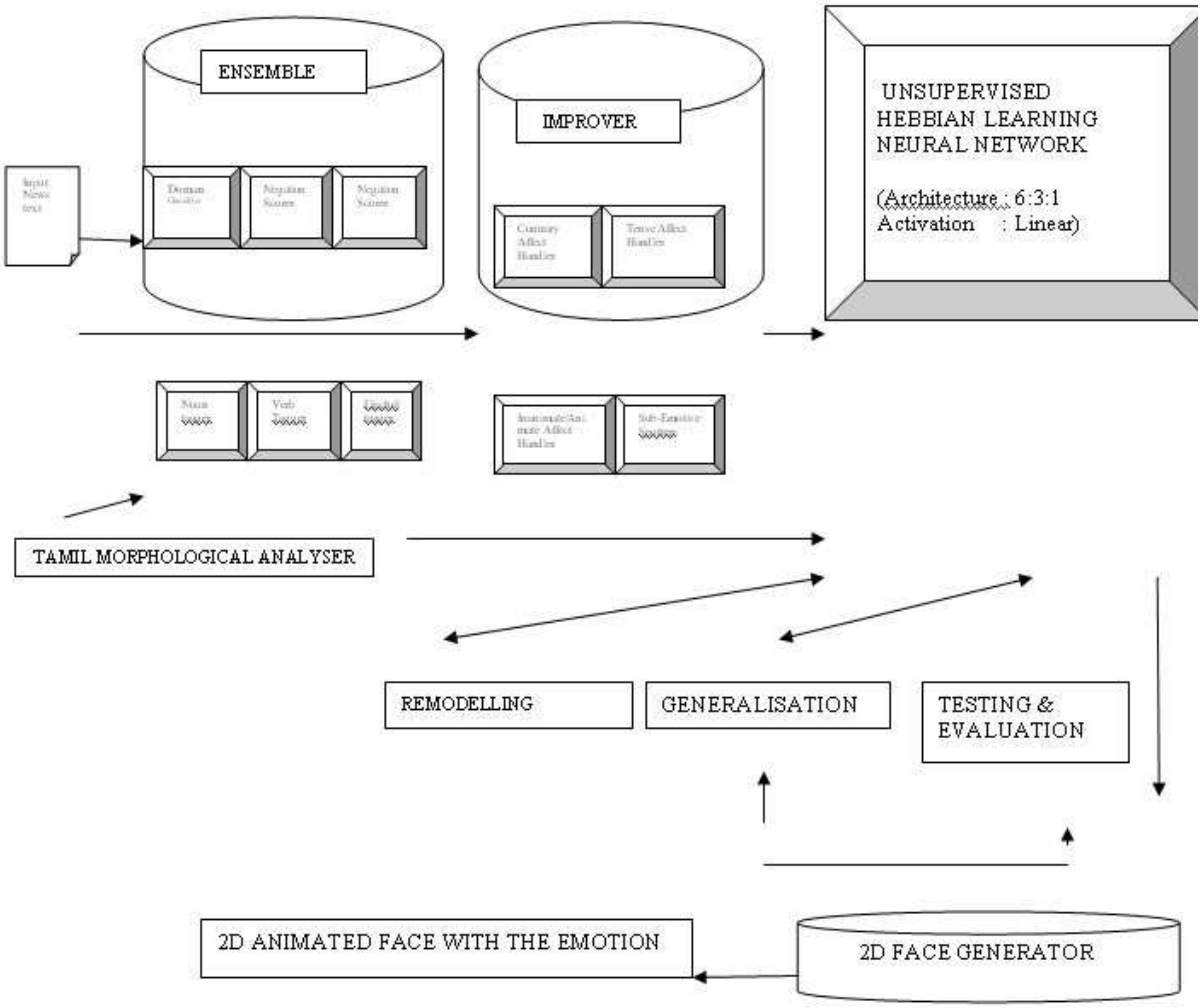


Fig 1: The Overall Architecture

A) The Domain Classifier

The Domain Classifier takes in documents which are already classified under five domains namely, politics, cinema, sports, business and health. Training process involves retrieving nouns and verbs using the Tamil Morphological Analyzer, calculating file constituent terms' domain frequencies and inserting them into the respective domain hash tables with terms as keys and term frequencies as values. Training reports a document's domain, based on the highest test domain frequency counter.

The algorithm is as follows

i) *Preprocessing:*

Let F_D represent the set of files in a domain and *f be* a file such that $f \epsilon F_D$. Let *T* be the bag of words in a domain without stop words and $t \epsilon T$.

1. **$\forall f \epsilon F_D$, remove stop words and tokenize**
2. **Get the nouns and verbs using the Tamil Morphological analyzer.**
3. **$\forall t \epsilon T$, $Ft = \sum f_t$ where f_t is the frequency of t in f.**

ii) *Training:*

Let H_D represent a domain hash table

$\forall t \epsilon T$, insert (t, H_D) where t being the key and the value v being

$$v = I_{occ} + \sum f_{dup(v)}$$

where I_{occ} is the first occurrence of the term in the domain and $f_{dup(v)}$ is the frequency of its duplicate.

iii) *Testing:*

Initialise domain frequency counters P_c, C_c, B_c, S_c and H_c. Given a document, remove the stopwords and tokenise.

Let Tok be the bag of nouns and verbs in the document, retrieved using the Tamil Morphological Analyser.

Convert Tok to a set by removing the duplicates.

$\forall t \epsilon$ Tok, get the frequencies of t, from all the domain hash tables. Increment the domain frequency counter of the domain that yields the highest frequency for t.

Report the domain of the test document as the domain that has the highest counter value.

*B) **The Negation Scorer***

The Negation classifier gets the documents, analyses whether positive words occur in the neighborhood of negative words or whether likes come close to dislikes and vice versa and assigns a score.

i) *Training:*

Let *F* denote set of all files in the corpus and let $f \epsilon F$. Let *T* be the bag of words in a file *f* without stop words and $t \epsilon T$.

a) $\forall f \epsilon F$, remove stop words and tokenize.

b) $\forall f \epsilon F$, let Neg denote the negation score of f primarily intialised to 1.

c) *$\forall t \epsilon T$, let i be the current position.Check whether t is a positive/negative word or like/dislike word.*

if t is positive/like, and a negative/dislike word occurs in a window of three places to the left or right, calculate Neg as

Neg = Neg – 0.01

if t is negative/dislike, and a negative/like word occurs in a window of three places to the left or right, calculate Neg as

Neg = Neg – 0.1

d) *If Neg < -0.5 ,report the document as negative.*

C) ***The Flow Scorer***

The Flow scorer assigns a score to each document based on the pleasantness of the words it has. The score is set in view of the phonetic classification in Tamil alongside with place and manner of articulation. Let *F* denote set of all files in the corpus and let $f \in F$. Let *T* be the bag of words in a file *f* without stop words and $t \in T$. Let t_g be the English equivalent of a Tamil word *t*.

a) $\forall f \in F$, *remove stop words and tokenize.*

b) $\forall t \in f$,*convert t to* t_g

c) $\forall t_g \in f$, *compute the flow score as the sum of the Maaththirai counts diminishing when Kurukkams appear.*

Table 1: *Kurukkams* and *Maaththirai*

Rule	Context	Final *Maaththirai*
KuttriyaLukaram	One among these குசுடுதுபுறு at the end of the word	1,(Decrease is 0.5)
Aukaarakkurukkam	ஔ in the beginning	1.5,(Decrease is 0.5)
Aikaarakkurukkam	ஐ in the beginning and middle	1.5,(Decrease is 0.5)
	ஐ in the end	1,(Decrease is 1)
Maharakkurukkam	வ before ம்	0.25,(Decrease is 1/4)

Table 2. Phonemes under Manner of articulation

Category	Manner of Articulation	Phoneme
Greater Rough	Retroflex, Trill	ட ற
Rough	Tap, Dental, Bilabial	க ச த ப ர
Intermediate	Semivowels, Approximants	ய ல ள ழ வ
Soft	Nasal	ங ஞ ண ந ம ன

Let *t* be the Tamil word, t_g be the Grapheme form and P_t be the bag of phonemes in *t* with $p \in P_t$. Let *GRscore* be the score of the greater rough category, *Rscore* be the score of the rough category, *Iscore* be the intermediate score and *Sscore* be the score of the soft category.

i) Calculate *GRscore*, *Rscore*, *Iscore*, *Sscore* as

$$GRscore = \sum f(p)_{GR}.$$

$$Rscore = \sum f(p)_{R}.$$

$$Iscore = \sum f(p)_{I}.$$

$$Sscore = \sum f(p)_{S}.$$

where $f(p)_{GR}$ is the frequency of a greater rough category phoneme , $f(p)_R$ is the frequency of a rough category phoneme, $f(p)_I$ is the frequency of a intermediate category phoneme and $f(p)_S$ is the frequency of a soft category phoneme.

ii) Calculate the *FinalRoughScore* and *FinalSoftScore* as

$$FinalRoughScore = GRscore + Rscore .$$

$$FinalSoftScore = Iscore + Sscore .$$

iii) If *FinalRoughScore* > *FinalSoftScore* $T \rightarrow$ *Pleasant*

iv) Else if *FinalSoftScore* > *FinalRoughScore* $T \rightarrow$ *Unpleasant*

v) Else $T \rightarrow$ *Neutral*

D) ***The Taggers***

At the start four taggers were constructed specifically, the noun tagger, verb tagger, *Urichol* tagger and case tagger. But for the instrumental and accusative cases, the left behind cases are not affective. For this reason, the constructed case tagger is unused as an input to the neural network. Nonetheless, the affect sensitive cases are incorporated in the affect specificity improver, namely the animate/inanimate affect handler.

Each tagger, picks up the respective part of speech in the document, analyses the major affect and tags the document with that affect. As a consequence, noun tagger reports the affect of nouns, verb tagger reports the affect of verbs and *Urichol* tagger reports the affect of *Urichols*. The generalized algorithm is as follows.

Let F denote set of all files in the corpus and let $f \in F$. Let T be the bag of words in a file f without stop words and $t \in T$. Let N denote nouns, V denote the verbs and U denote the *Urichols* in a file. Let $n \in N$, $v \in V$, $u \in U$.

a) $\forall f \in F$, *remove stop words and tokenize.*

b) $\forall f \in F$, *get N for Noun Tagger, V for Verb tagger and U for Urichol tagger, using the Tamil Morphological analyzer.*

c) $\forall n \in N$, *use the noun affect lexicon to determine the noun affects. Initialize respective noun affect counters for the basic six emotions and increment them depending on the affect of each n.*

d) $\forall v \in V$, *use the Verb affect lexicon to finalize the verb affects. Initialize respective affect verb counters for the basic six emotions and increment them depending on the affect of each v.*

e) $\forall u \in U$, *find the Urichol affect using the Urichol affect lexicon. Initialise respective affect counters for the basic six emotions and increment them depending on the affect of each u.*

f) *For each tagger, report the final affect of a file f, as the affect of the respective affect counter that has the maximum value.*

E) ***The Tense Affect Handler***

a) $\forall f \in F$, *get the verbs under each of the three tenses. Let Pr represent the present tense, Pa denote the past tense and F denote the future tense.*

b) Analyze the affect of each tense category verbs.

c) Prioritize Present, Future and then Past tenses.

d) Report the high frequent affect of the Present tense. If present tense verbs are absent, report the maximum frequent affect of future tense, else report the high frequent affect category of the past tense.

F) The Inanimate/Animate Handler

a) $\forall f \in F$, *get the verbs using the Morphological analyzer.*

b) *Restrict a window size of one to the left to find whether affect sensitive cases are present.*
c) *Analyze the affect of the verbs and categorize them under mild and dangerous.*
d) *Analyze the cases and see if they add to the affect of the verbs or nullify it.*
e) *If affect intensity is increased report danger, else report the affect implied.*

G) The Contrary Affect Handler

a) *∀f ϵ F, check whether there are multiple entities in a sentence.*
b) *Using the Profile monitor, check whether the entities are in like list or dislike list.*
c) *Analyze the nouns next to the entities to see whether they are favorable to the preferred entities or not.*
d) *Report the affect as the affect of the nouns with reference to the preferred entity.*

H) The Sub-emotion Spotter

a) *∀f ϵ F, Get all nouns, verbs and urichols using the Tamil morphological analyzer.*
b) *Use the constructed sub-emotive affect lexicon to identify the high frequent sub emotion.*
c) *Report the maximum occurring affect based on the sub- affect.*

I) ***The Neural Network***

Initially a supervised Backpropogation network was constructed with the architecture 6:3:1. However, since emotion recognition is to do with emotional intelligence, unsupervised learning was preferred and Hebbian learning was incorporated. The forget factor is fixed as 0.02 and the learning factor as 0.1. Following is the pseudo code.

a) *∀f ϵ F, get the outputs from Domain Classifier, Negation Scorer, Flow Scorer and the three affect taggers.(The Improver modules fail to aid emotion recognition by getting eclipsed toward certain domains and are hence overlooked.)*
b) *Initialize the network weights and threshold values which are set in the range of 0 to 1.*
c) *Start the learning for each file using Hebb's rule.*
d) *Report the affect as the affect of the output activation that approximately equals an affect value.*
e) Stop the learning when steady state is embarked or sufficient number of epochs has been elapsed.

IV. Results and Evaluation

Both Batch and Sequential learning are supported. The precision of the Neural network is 60% . Other datasets used were lyrics and stories. Stories usually have sub plots and hence overall affect usually gets eclipsed towards neutral. Hence, news (400 files) and lyrics (230) were used for training. Validation set included 50 news files with equal contribution from five domains and 10 lyrics.

For lyrics and stories, the neural net has a better precision of 70%. The precision gets increased when epochs increase for all the datasets. Simple supervised Back propogation network was implemented and used as the Baseline whose precision was 30% more than the constructed even with minimum epochs.

Following is a graph that depicts how precision of emotion found varies with respect to the number of epochs, in the case of a single news text. Values nearer to zero indicate the drastic variation of the reported affect from the actual one (ie joy instead of sad) and values near 100 indicate the closeness towards the actual affect.

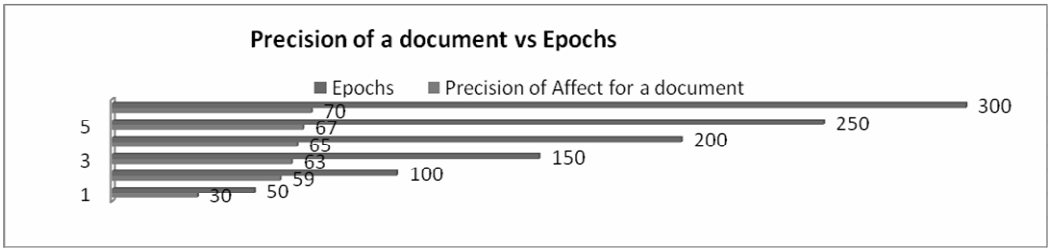


Fig 2: Epochs vs Precision of affect of a document.

The neural network suffers from ambiguity problems in the case of closely related categories like love-joy and sad-fear irrespective of dataset. The supervised Back propogation network is used as the baseline which is 30% more in precision than the Unsupervised Hebbian Learning.

Besides, the Unsupervised Hebbian learning Neural net has exponential complexity and consumes two thirds of the physical memory. The affect reporting of a single file amounts to three minutes where approximately one and a half minute is taken for indexing and loading of the domain hash tables by Domain Classifier. The CPU usage during the learning varies roughly from 11% to 72%.

V. Conclusion

Thus, an unsupervised Hebbian learning neural network is constructed which fetches its major inputs from a domain classifier, two sentimental scorers and three part of speech taggers to figure out the affect in the presented text. As is the case with almost all natural language processing applications, ambiguities do exist in emotion recognition; here among closely related affective categories like love-joy and sad-fear. However, a competitive strategy in unsupervised learning can be opted to resolve this issue, as such a learning is concerned with demarcating one from the rest. Identification of other affect sensitive features could further aid in precise emotion detection.

VI. References

- Ahmed Abbasi, Hsinchun Chen, Sven Thomas, Tianjun Fiu, "Affect Analysis of Web forums and blogs using correlation ensembles", IEEE Transactions on Knowledge and Data Engineering,Vol.20, No.9, pp.1168-1180, Sepetember 2008.
- Edward Chao-Chun Kao, Chun Chieh Liu, Ting-Hao Yong, Chang Tai Hseih, Von Wun Su, " Towards text-based Emotion detection", International Conference on Information Management and Engineering,2008.
- Sowmiya, Madhan Karky V, "Face Waves : 2D Facial Expressions based on Tamil Emotion Descriptors", World Classical Tamil Conference, June 2010.

- Ze-Jing Chuang and Chung-Hsien Wu, "Multimodal Emotion Recognition from Speech and text", Computational Linguistics and Chinese Language Processing, Vol.9,No.2, pp. 45-62, August 2004.
- Maja Pantic, Leon J.M. Rothkrantz, "Toward an Affect-Sensitive Multimodal Human-Computer Interaction", Proceedings of the IEEE, Vol.91, pp.1370-1390, September 2003.
- Yong-Soo Seol, Dong-Joo Kim and Han-Woo Kim, "Emotion Recogniton from Text Using Knowledge-based ANN", The 23rd International Technical Conference on Circuits/Systems, Computers and Communication 2008.
- Donn Morrison, Ruili Wang, W.L.Xu, Liyanage C.De Silva, "Voting Ensembles for Spoken Affect Classification", Elsevier Science, February 6,2006.
- Futoshi Sugimoto, Yoneyama Masahide, "A method for classifying Emotion of Text based on Emotional Dictionaries for Emotional Reading".
- J.Nicholson, K.Takahashi, R. Nakatsu, "Emotion recognition in Speech using Neural networks", Neural Computing and Applications, Springer-Verlag London limited, pp.290-296, 2009.
- Lyle N.Long, Ankur Gupta, "Biologically –Inspired spiking Neural networks with Hebbian learning for vision processing", AIAA 46th Aerospace Sciences meeting, Reno, NV, Jan 2008.
- Lei Shi, Bai Sun, Liang Kong, Yan Zhang, "Web forum Sentiment analysis based on topics", IEEE Ninth International Conference on Computer and Information Technology, 2009.
- Changrong Yu, Jiehan Zhou, Jukka Riekki, "Expression and analysis of Emotions – A Survey and Experiment", IEEE Symposia and Workshop on Ubiquitous, Autonomic and Trusted Computing, 2009.
- Irene Albrecht, Jorg Haber, Kolja Kahler, Marc Shroeder, Hans Peter Siedel, "May I talk to you :-) Facial Animation from Text", IEEE Proceedings of the 10th Pacific Conference on Computer Graphics and Applications, 2002.
- Aysegul Cayci, Selcuk Sumengen, Cagatay Turkey, Selim Balcisoy, Yucel Saygin, "Temporal Dynamics of User interests in Web search queries", International Conference on Advanced Information Networking and Application Workshops, 2009.
- Tim Andersen, Wei Zhang, "Features for Neural net based region identification for Newspaper documents", Proceedings of the seventh International Conference on Document Analysis and recognition, 2003.
- Taeho Jo, Malrey Lee, Thomas M Gatton, "Keyword Extraction from Documents using a Neural network model", International Conference on Hybrid Information Technology, 2006.
- Azam Rabiee, Saeed Setayeshi, "Persian Accents Identification Using an adaptive Neural network", IEEE Second International Workshop on Education Technology and Computer Science, 2010.
- Changua Yang, Kevin Hsin-Yih Lin, Hsin-His Chen, "Writer meets Reader: Emotional Analysis of Social Media from both the Writer's and Reader's perspectives", IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Workshops", 2009.

Tamil Online handwriting recognition using fractal features

Rituraj Kunwar and A G Ramakrishnan
MILE Lab, Dept of Electrical Engineering, Indian Institute of Science, Bangalore.

Abstract

We present a fractal coding method to recognize online handwritten Tamil characters and propose a novel technique to increase the efficiency in terms of time while coding and decoding. This technique exploits the redundancy in data, thereby achieving better compression and usage of lesser memory. It also reduces the encoding time and causes little distortion during reconstruction. Experiments have been conducted to use these fractal codes to classify the online handwritten Tamil characters from the IWFHR 2006 competition dataset. In one approach, we use coding and decoding process. A recognition accuracy of 90% has been achieved by using DTW for distortion evaluation during classification and encoding processes as compared to 78% using nearest neighbor classifier. In other experiments, we use the fractal code, fractal dimensions and features derived from fractal codes as features in separate classifiers. Whereas the fractal code was successful as a feature, the other two features are not able to capture the wide within-class variations.

Introduction

Fractal codes are the compressed representation of patterns, based on iterative contractive transformations in metric spaces proposed by Barnsley. A simplified version of the fractal block coding technique for digital images has been used to encode the 1-D ordered online handwritten character patterns. A novel partitioning algorithm has been proposed to reduce the computation complexity of encoding and decoding, with a minor fall in recognition accuracy.

Building fractal codes for handwritten characters

We need to find the collection of affine transforms of the online handwritten character. The raw online handwritten character is first preprocessed using three steps: (i) smoothing (ii) re-sampling the variable number of points in each character to 60 points. (iii) normalizing the x and y coordinates between 0 and 1. The handwritten character locus is divided into non-overlapping range segments. Each range segment has a fixed number of points (R) in it. Last point of each range becomes the first point of the next range, except in the case of the last range.

Creating a pool of domain segments

The domain pool is formed for each character locus. Domain pool is the collection of all possible domain segments. The number of points in each domain segment is chosen to be double that in each range segment, D = 2R. Domain pool can be obtained by sliding the window containing D points at a time. The window is first located at the beginning of the stroke. The window is moved along the

stroke by δ points, in such a way that it does not cross the end point of the stroke. The step δ has been chosen as R/2 in our experiments.

Constructing transformed Domain pool

Transformed domain pool is constructed by multiplying each domain segment with the eight isometrics that involve reflection and rotation about different axes. To begin with, each domain is translated to its centroid and scaled down by the contractivity factor (s=0.5). The following transformations are then applied to each of the candidate domain segment.

$$\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\cdot\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\cdot\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\cdot\begin{bmatrix}0 & 1\\ -1 & 0\end{bmatrix}\cdot\begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix}\cdot\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}\cdot\begin{bmatrix}-1 & 0\\ 0 & -1\end{bmatrix}\cdot\begin{bmatrix}0 & -1\\ -1 & 0\end{bmatrix}$$

The above transformations produce a whole family of geometrically related domain segments. In domain pool, matching blocks will be looked for encoding the online handwritten character.

Searching for the most similar domain segment for each range: Each affine transformed domain segment is re-sampled into R points and then its centroid is translated to that of the concerned range segment. Distance between them is found. Similarly, distances w.r.t to the all the domain segments is calculated. The most similar domain segment corresponding to each range segment is identified and fractal code is stored corresponding to the particular range. The fractal codes are similarly obtained for all the online handwritten characters.

Fractal codes corresponding to each range segment consists of (1) the range segment index, (2) the range segment centroid, (3) index of the most similar domain segment and (4) the index of transformation used out of the 8 transformations.

Issue related to constructing fractal codes

The whole character is divided into range segments of equal number of points. Smaller the number of points in each range segment, the more minutely we can capture the complexity in any region of the character. The number of range segments per character is thus inversely proportional to the number of points in each range. Again, the encoding speed is inversely proportional to the number of range segments per character. It has been noted that there are certain region in a character where the curliness is minimal so in those area the range segment size could be increased still encoding the region precisely.

Steps to encode a handwritten character where the number of points in each range is variable:

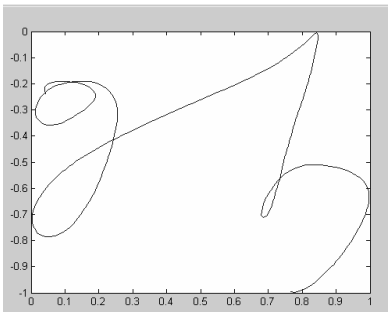


Fig 1. Tamil handwritten character 'aa'

Cumulative angle 'θ_C' is calculated starting from the first point and traversing the character stroke till it crosses an empirically set threshold of θ_T. Smaller the threshold, finer is the encoding. Figure 1 shows a sample of the handwritten character /aa/ in Tamil. Figure 2 shows the effect of the choice of the angle threshold on the reconstruction error.

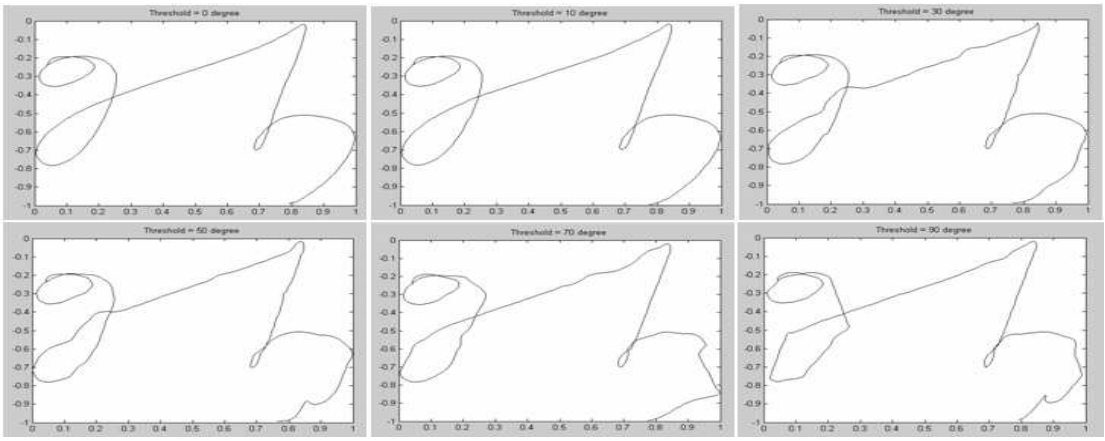


Fig 2. Illustration of the distortion in reconstruction with different angle thresholds ϕ_T. Here reconstruction is performed from the fractal codes of the character /aa/ shown in Fig 1. For encoding, different values of range sizes are used, namely, 4, 8, 12, 16 and 20.

Algorithm for partitioning

1. Domain pool of different sizes (namely 8, 16, 24, 32, 40) are constructed corresponding to the range sizes of 4, 8, 12, 16 and 20**.** By size, we mean the number of points in each domain.
2. Start from the first point and move along the character from one point to the next and calculate the cumulative change in angle $\boldsymbol{\theta_C}$.
3. The No. of points (K) till the point penultimate to the one, where $\boldsymbol{\theta_C}$ crosses the threshold $\boldsymbol{\phi_T}$ is noted.
4. The range size closest to and less than K is chosen. The most suitable domain is chosen from the corresponding domain pool and the fractal codes are stored.
5. The last point of the present range is then considered as the first point of the new range and the process repeats starting from step 2.

Along with the fractal codes, the size of the range chosen is also stored. If the end point is reached with $\boldsymbol{\theta_C} < \boldsymbol{\phi_T}$, then step 4 is followed, where K includes the last point also since $\theta C < \phi T$). If at the end, few points are left which is less than the smallest range, they are discarded else step 4 is repeated.

Algorithm for reconstruction: Banach's contractive mapping theorem states: "If a contractive mapping 'W' (which are the fractal codes here) is defined, then iterative application of the mapping on any sequence of the same space will lead to a Cauchy's sequence which will converge to a fixed, unique point.

CASE I: Range having fixed number of points

A random initial pattern having the same number of points is taken or generated. A domain pool of size double that of the range is created in a manner similar to the encoding process. First fractal code is

taken corresponding to the first range of the pattern, and operations are performed on the corresponding domain indicated by the domain index in the code of first range. The indicated domain segment's origin is shifted to its origin and then it is scaled down by the contractivity factor (0.5 here). Then the affine transformation as indicated in the code is applied on the scaled domain segment. Finally the transformed domain's centroid is shifted to the range segment centroid as present in the fractal code. The above steps are repeated to decode all the range segments. Then the whole decoded locus is smoothened. The above steps are repeated till the termination condition is satisfied to finally converge to a fixed and unique pattern. Termination condition could be (i) an empirically set fixed number of iterations, sufficient for convergence or (ii) minimal or no distortion between two consecutive patterns produced by 2 consecutive iterations.

CASE II: Range having variable number of points in each range

In this case, multiple domain pools are created out of the random pattern taken for the reconstruction. Using the extra information given in the fractal code pertaining to the range size to be chosen so that domain segment is picked up from the appropriate domain pool. The rest of the steps are same as the case for the reconstruction with fixed range size. Using above two methods, fractal codes of any give pattern can be created and the same pattern could be decoded using any random pattern after applying this reconstruction algorithm iteratively for few times.

1. Classification by using fractal codes in construction and reconstruction:

The above fractal encoding and decoding method has been used for classification of characters. The assumption behind this classification is that if a sample of a class (say /a/) is encoded and fractal codes are obtained. The following process of reconstruction if started in 2 ways i.e. firstly by applying the reconstruction algorithm iteratively on a random pattern of any different class (anything other than 'A') and secondly doing the same on any random pattern of the same class (i.e. 'A') then the distortion between the initial pattern and the pattern obtained after first iteration of reconstruction is relatively much more in the first case than in the second. The reason behind this is that reconstruction process leads to convergence to the pattern whose code is used for reconstruction. And since the class of the initial pattern in the second case and the fractal code is same, the distortion in the second case is smaller than the first case.

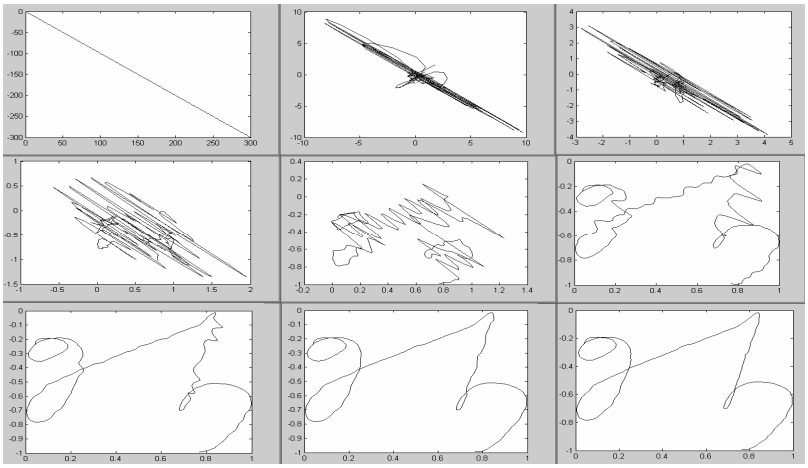


Fig 3. In the above image, reconstruction process is shown which starts from a random straight line and finally converges to a pattern which is very close to the original pattern after 8 iterations. The original pattern (in Fig 1) was encoded using different range sizes of 4, 8, 12, 16 and 20 with the threshold angle of 30 degrees.

Character classification using fractal codes:

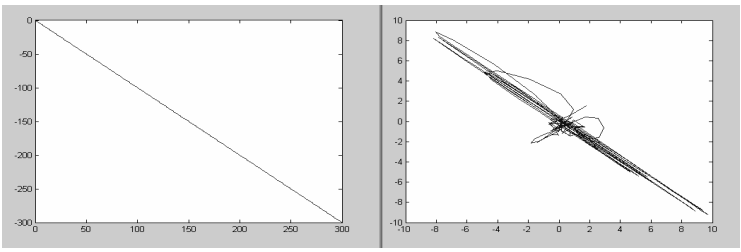


Fig. 4. The above image shows the distortion created, when an iteration of reconstruction was performed. This shows that the distortion is huge if the starting pattern (above left) is very different from the original pattern, whose fractal codes are used for reconstruction (in this case Fig 1).

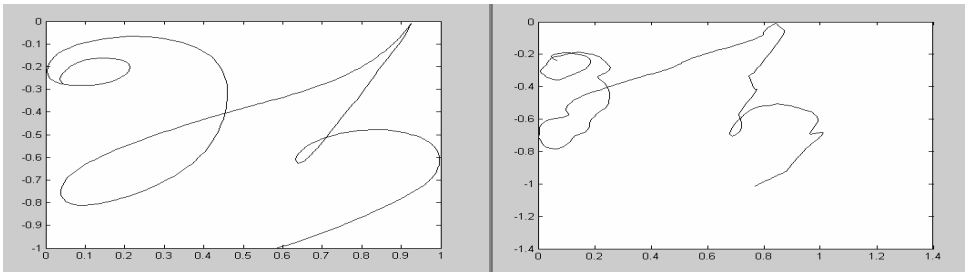


Fig. 5. The above image shows the distortion created, when an iteration of reconstruction was performed. This shows that the distortion is much less if the starting pattern (above left) is not very different from the original pattern, whose fractal codes are used for reconstruction (in this case Fig 1).

Classification Algorithm

In the present research, fractal codes of 'N' samples of entire 156 classes are computed and stored. To classify a test sample, following steps are taken.

a) An iteration of the reconstruction algorithm is applied on the test sample using the fractal code of each sample of each class.
b) The distortion 'D' is calculated between the initial pattern and the pattern obtained after one step of reconstruction. Thus the distortion matrix of size 156*N is obtained.
c) The row number of the minimum value of the distortion is found from the distortion matrix and assigned to the test sample.

Distortion evaluation:

The distortion between two patterns can be evaluated by finding the distance between them. The distance between 2 patterns is measured by Nearest Neighbor (NN) method. The issue with NN is that the matching is done point by point which increases the distance unusually and thus decreases the classification accuracy (evident from the result table). This drawback in distance evaluation of NN

is addressed by DTW pattern matching which is more intuitive. This intuitive matching takes place because DTW matches similar subsections of the patterns thus producing a reasonable distance between patterns. This method hence produces a remarkable increase in accuracy as shown Table 1.

Classification using the fractal codes as a features in the Nearest Neighbor (NN)

In this method, the fractal codes are used as features and fed into a NN classifier. An accuracy of approximately 65% is obtained.

Classification using fractal dimension or features derived from the fractal codes

Fractal dimension is a unique identity of any pattern or object. However, because of the mere nature of handwritten character recognition (i.e. large variation within every class), it fails completely to classify any random sample. The features derived from the fractal codes like MMVA and DRCLM, though successful in problems like face recognition and signature verification, fail in recognizing handwritten characters.

Results

Table 1: Results show how the DTW comparison during reconstruction impacts the recognition accuracy

Fixed Range Size	DTW used?	No. of Training samples	No. of Testing samples	Accuracy (in %)
4	No	20	50	78.9
4	Yes	20	50	90.4

Table 2: Results show the efficacy of the Partitioning algorithm in improving the efficiency of the recognition system (in terms of time) with marginal drop in accuracy. Variable range sizes used (4, 8, 16, 24 and 32). No. of training and testing samples used are 5 and 30, respectively, No. of classes used is 156.

Threshold angle (degree)	DTW used for distortion evaluation?	Encoding time per sample (sec)	Accuracy (in %)
0	Yes	31	90.44
10	Yes	22	86.43
30	Yes	12	85.04
50	Yes	10	83.55
70	Yes	8	81.60
90	Yes	6	80.87

Acknowledgment: The authors thank Technology Development for Indian Languages (TDIL), DIT, Government of India for funding this research, as part of the research consortium on Online handwriting recognition of Indian languages.

References

- M. F. Barnsley, Fractals everywhere, New York: Academic, 1988.
- T. Tan and H. Yan, Face recognition by fractal transformations, IEEE ICASSP, 6:3405-3408, 1999.
- Mozaffari S., Faez K. and Faradji F, One Dimensional Fractal Coder for Online Signature Recognition, IEEE ICPR, 2:857- 860, 2006.

Neuroscience inspired segmentation of handwritten words

A G Ramakrishnan and Suresh Sundaram
MILE Lab, Dept of Electrical Engineering, Indian Institute of Science, Bangalore.

The challenge of segmenting online handwritten Tamil words has hardly been investigated. In this paper, we report a neuroscience-inspired, lexicon-free approach to segment Tamil words into its constituent symbols (recognizable entities). Based on a simple dominant overlap criterion, the word is grossly segmented into candidate symbols (stroke groups). However, this segmentation is not fully reliable because of varying writing styles resulting in varying levels of overlap. Taking cues from vertebrate visual perception, we utilize both feature based attention and feedback from the classifier to detect possible wrong segmentations. This attention-feedback segmentation (AFS) strategy splits or merges the stroke groups to correct the segmentation errors and forms valid symbols. This maiden attempt on segmentation is tested on 10000 handwritten words collected from hundreds of writers. The efficacy of AFS in segmentation and improving the recognition performance of the handwriting system is amply demonstrated. Our results show a segmentation accuracy of over 99% at symbol level.

Need for segmenting handwritten words

Since attempts to segment cursively handwritten English words have largely failed, researchers working on Indic scripts too feel that it is not advisable to try to segment individual characters from handwritten documents. However, we firmly believe that it is not only possible, but also something that ought to be done, if one is interested in recognizing words such as proper names appearing in the name and address fields of handwritten forms. Thus, this opens up the possibility of developing a recognizer that can handle unrestricted vocabulary, including any unusual word of foreign origin, such as names of people or places from other countries. Thus, we believe that our work is the first of its kind in proposing an approach for handwriting recognition that does not limit the writer from writing any text of any origin.

Motivation for Attention-Feedback Segmentation

Traditional pattern recognition [1, 3-5, 8, 10, 13-15] primarily follows a feedforward architecture, whereas the same in mammalian brain involves complex feedback structures. Studies on visual perception in primates demonstrate the effect of attention on the response of the visual neurons [2]. Feature based attention biases the neuronal responses as though the attended stimulus was presented alone. Also, shifting spatial attention from outside to the inside of the receptive field increases the neuronal responses. Motivated by these observations, we incorporate local feature based attention to correct and improve segmentation [9]. Further, studies on visual pathways show extensive feedback from the cortex to the lateral geniculate nucleus (LGN), which have both inhibitory and facilitatory effects on the responses of LGN relay cells. In our work, we use feedback based on features as well as from the classifier posterior probabilities to rectify any incorrect segmentation by regrouping the strokes. Thus, we call our approach as 'attention-feedback' strategy for segmentation.

Further, studies on scene perception by humans [6] indicate that visual processing follows a top-down approach. The global cues characteristing the visual object, that appear within the visual span, are perceived before the local features. The human perceptual system treats every scene as if it were in the process of being focussed or zoomed in on, whereas initially, it is relatively less distinct. Moreover, the human perceptual processor has the capability to select parts of the input stimulus that are worth to be paid attention to. Motivated with these observations from the field of neuroscience, we present a segmentation strategy that first works on the global feature of overlap to output candidate Tamil stroke groups for the given input strokes. By analyzing local features characteristic to the given input pattern, we reevaluate the segmentation and modify the segmentation when found necessary. The localized features are derived by zooming on paying attention to specific parts of the online trace. Essentially, we adopt a multi-pass system, wherein fine grained processing is guided by the prior cursory (global) processing.

Data used for the study

The 155 distinct Tamil symbols (comprising 11 vowels, 23 base consonants, 23 pure consonants, 92 CV combinations and 6 additional symbols) are presented in Appendix A. The publicly available corpus of isolated Tamil symbols (IWFHR database) is used for learning various statistics about Tamil symbols. The primary focus of this work is to address the challenges of segmentation. Towards this purpose, Tamil words are collected using a custom application running on a tablet PC and saved using a XML standard [7]. High school students from across 6 educational institutions in Tamil Nadu contributed in building the word data-base of 100, 000 words, referred to as the 'MILE Word Database' in this work [12]. Out of these, 10,000 words are used for this study. The words have been divided into 40 sets, each comprising 250 words. Owing to the comparable resolution of our input device to that used in the IWFHR dataset, statistical analysis performed on the symbols in the IWFHR database are applicable to the Tamil symbols in the MILE word database.

Dominant Overlap Criterion Segmentation

An online word can be represented as a sequence of n strokes $W = \{s_1, s_2, \ldots, s_n\}$. In the case of multi-stroke Tamil symbols, strokes of the same symbol may significantly overlap in the horizontal direction. The word is first grossly segmented based on a bounding box overlap criterion, generating a set of stroke groups. In this 'Dominant Overlap Criterion Segmentation' (DOCS), the heavily overlapped strokes are merged. A stroke group is defined as a set of consecutive strokes merged by the DOCS step, which is possibly a valid Tamil symbol.

For the k-th stroke group S_k under consideration, its successive stroke is taken and checked for possible overlap. Significant overlap necessitates the successive stroke to be merged with the stroke group S_k. Otherwise, the successive stroke is considered to begin a new stroke group S_{k+1}. The algorithm proceeds till all the strokes of the word are exhausted.

Neuroscience-inspired segmentation

The stroke groups obtained from the above dominant overlap criterion segmentation are preprocessed by smoothing, normalization and resampling into standard number of equi-arc length spaced points.

The x and y coordinates of these processed stroke groups and their first and second derivatives are used as features for recognition using a support vector machine (SVM) classifier that outputs class labels and their posterior probabilities. Obviously, DOCS being simple, does not always result in correct segmentation. Sometimes it results in over segmentation of a single multi-stroke character into two stroke groups; other times, two distinct characters get combined into a single stroke group, due to the way they are written.

Attention Features

Figure 1 shows the complete block schematic of the proposed segmentation scheme. An over-segmented symbol is usually small and hence results in low aspect ratio as well as has very few dominant points (points where the curvature is high). By paying attention to these features extracted from the stroke groups output by DOCS block, one can suspect wrong segmentation. Further, the symbols that result from over- or under-segmentation are classes that the classifier has not come across. Thus, these symbols usually result in a low confidence level of the classifier. Thus, the posterior probability of the classifier, when fed back to the input stages, can be used to invoke the computation of the attention features. The feedback, together with the attention features suggest possible resegmentation of the input strokes, resulting in new possible stroke groups. These modified stroke groups based on merger or splitting of original stroke groups, are once again recognized by the classifier after preprocessing and extraction of recognition features. An improved posterior probability of the new stroke group confirms right segmentation. Thus, the refinement in segmentation is caused based on memory, attention and feedback mechanisms prevalent in human perception. We call this as "attention-feedback segmentation (AFS)".

Commonly found segmentation issues

The two Tamil characters that ought to have a minimum of three strokes are the long /i/ (nedil) and the aydam. Since in both of these cases, in general there is no overlap between the final dot and the rest of the character, they always are over segmented into two or more stroke groups.

Pure consonants (*mey ezhuthu*), when they are written with the dot (*pulli*) beyond the base consonant, result in over segmentation too.

Characters such as /ka/, /nga/ and /ra/, which start with an initial vertical segment, are written by many with multiple strokes, with the first stroke being a simple down-going vertical line. These characters have a potential to be over segmented, if the following part of the character does not clearly overlap with the vertical line.

All CV combinations of /i/ and /I/ and the CV combinations of /u/ and /U/ with borrowed consonants such as /ja/ and /sha/ also have a tendency to be over-segmented, if the vowel matra is written with no horizontal overlap with the consonant.

Under segmentation occurs if the ending part (usually bottom extensions of /ta/ or /Ra/) of the following character goes far left below the previous character, causing significant horizontal overlap between them. At other times, people write two successive characters so closely, that there is significant overlap between them.

Naturally, in all the above cases, the simple segmentation (DOCS) is likely to result in wrong segmentation leading to erroneous recognition results.

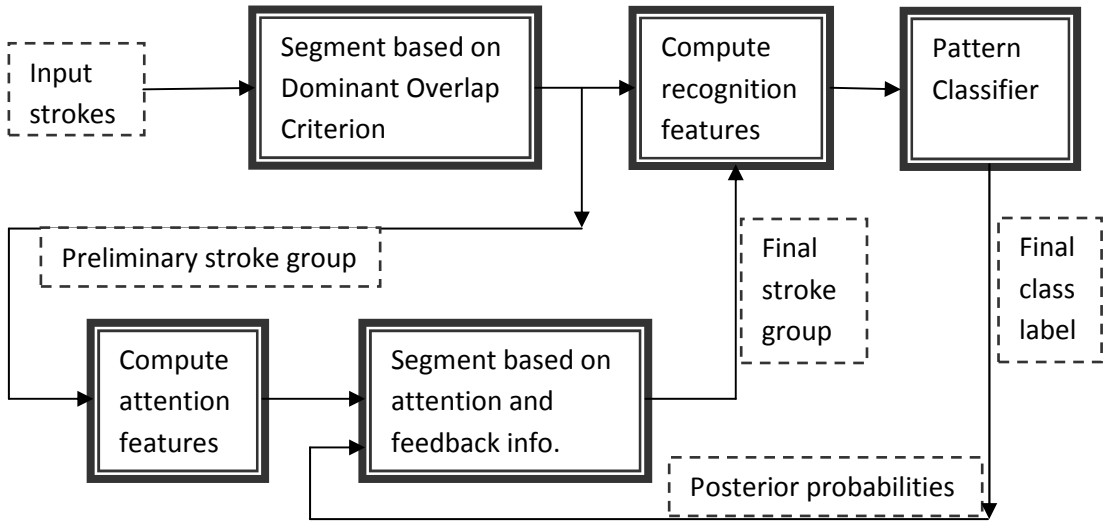


Fig. 1. Block diagram of neuroscience inspired segmentation of Tamil handwritten word [9].

Segmentation results on the MILE Tamil Word Database

The proposed techniques are tested on the subset of 10,000 words. However, to start with, we evaluate the performance on a set of 250 words (denoted as DB1), that has a significant number of errors resulting from the DOCS. Of the 103 errors, 89 (or 86%) correspond to the merging of valid symbols, and the rest, to broken symbols. The AFS module aids in properly detecting and correcting 91 (or 90%) of these errors. In addition, the methods proposed effectively merge 11 (or 78%) of the over-segmented stroke groups to valid symbols. The improvement in character segmentation rate in turn reduces the number of wrongly segmented words. It is observed that only 7 of the total 250 words remain wrongly segmented after the AFS scheme, as against 67 words after the DOCS scheme. On evaluating the performance across the database of 10000 words, we obtain a 86% reduction in character segmentation errors.

Recognition results on the MILE Database

We report experimental results demonstrating the impact of the proposed AFS strategy on the recognition of symbols in the MILE word database. Since a significant percentage of DOCS errors are corrected by AFS, a drastic improvement of 16% (from 70.5 % to 87.1 %) in symbol recognition is observed. In computing the symbol recognition rate, apart from the substitution errors, we take into account the insertion and deletion errors, caused by over-segmentation and under-segmentation, respectively. The edit distance is used for matching the recognized symbols with the ground truth data. Moreover, 11.6 % of the words, (29 additional words) wrongly recognized after DOCS, have

been corrected by the proposed technique. Across the 10000 words in the MILE Word database, an improvement of 4% (from 83 to 87%) in symbol recognition rate has been obtained.

Conclusion

In this paper, we present a maiden attempt based on significant feedback from the classifier to the input blocks such as feature extraction and segmentation, as well as the use of memory (prior knowledge) to result in a very effective segmentation of online handwritten Tamil words. This approach being general, can be extended to any other Dravidian script, as well as any other script where cursive writing is not practiced. To our knowledge, there is no reported systematic research work on segmenting the individual characters or recognizable standard symbols from online handwritten words for any Indic language that does not have the *shiro rekha* (head line). Thus, we are unable to compare the performance of our work with any other technique. However, the results are promising and have also led to improved recognition of the handwritten words [16], thus confirming the possibility of proper segmentation of online Tamil words. We intend to extend this work very soon to online Kannada handwritten words.

Acknowledgment

We thank Ms. Nethra Nayak, Mr. Rituraj Kunwar, Ms. Archana C P, Mr. Shashi Kiran, Ms. Chandrakala, Mrs. Shanthi Devaraj, Ms. Saranya and Ms. Sountheriya for their efforts in data collection and annotation, which made these experiments possible. We thank Dr. Arun Sripathi of the Centre for Neuroscience, IISc for the very useful discussions we had on visual perception. Special thanks to Technology Development for Indian Languages (TDIL), Department of Information Technology (DIT), Government of India for funding this research, as part of a research consortium on online handwriting recognition in several Indic scripts. We thank Prof. Deivasundram (University of Madras), AVM Matriculation Higher Secondary School Virugambakkam, Chennai, Govt. Boys Higher Secondary School, Sulur, Presidency College, Triplicane, Chennai and IIT Madras for contributing to the data set. We also thank Dr. Anoop Namboodiri for giving us the word level annotation tool. We thank CDAC, Pune for being a partner in finalizing the XML standard for handwritten data collection in Indic languages.

References

- N Joshi, G Sita, A G Ramakrishnan, S Madhavanath, "Comparison of elastic matching algorithms for online Tamil handwritten character recognition", Proc. IWFHR (2004) 444-449.
- G M Boynton, "Attention and visual perception. Current Opinion in Neurobiology", (15) (2005) 465-469.
- H Swethalakshmi, C Chandra Sekhar, V S Chakravarthy, "Spatiostructural features for recognition of online handwritten characters in Devanagari and Tamil scripts", ICANN (2) (2007) 230-239.
- A Bharath, S Madhvanath, "Hidden markov models for online handwritten Tamil word recognition", Proc. ICDAR (2007) 506-510.
- Amrik Sen, G. Ananthakrishnan, Suresh Sundaram, A. G. Ramakrishnan, "Dynamic space warping of strokes for recognition of online handwritten characters. IJPRAI (2009) 23(5): 925-943.

- Arun P Sripati and Carl R Olson, "Representing the forest before the trees: a global advantage effect in monkey inferotemporal cortex", The Journal of Neuroscience, June 17, 2009, 29(24):7788 –7796.
- Swapnil Belhe, Srinivasa Chakravarthy, A. G. Ramakrishnan, XML standard for Indic online handwritten database, ACM - Proceedings of the International Workshop on Multilingual OCR, 2009.
- M. Mahadeva Prasad, M. Sukumar, A. G. Ramakrishnan, Divide and conquer technique in online handwritten Kannada character recognition, ACM - Proc International Workshop on Multilingual OCR, 2009.
- Suresh Sundaram and A G Ramakrishnan, "Verification based segmentation approach for online words", Indian Patent Office Ref. No. 03974/CHE/2010.
- Shashi Kiran, Kolli Sai Prasada, Rituraj Kunwar, A. G. Ramakrishnan, "Comparison of HMM and SDTW for Tamil handwritten character recognition", Proc. 2010 IEEE International Conf Signal Processing & Communication.
- Rituraj Kunwar, Mohan P., Shashi Kiran, A. G. Ramakrishnan, "Unrestricted Kannada online handwritten akshara recognition using SDTW, Proc. 2010 IEEE International Conf Signal Processing & Communication.
- B Nethravathi, C P Archana, K Shashikiran, A G Ramakrishnan, V Kumar, "Creation of a huge annotated database for Tamil and Kannada OHR", Proc. IWFHR (2010) 415-420.
- M. Mahadeva Prasad, M. Sukumar, A. G. Ramakrishnan, "Orthogonal LDA in PCA transformed subspace", Proc. 12th International Conf Frontiers in Handwriting Recognition (ICFHR 2010), Nov 2010.
- Venkatesh N, A G Ramakrishnan, "Choice of classifiers in hierarchical recognition of online handwritten Kannada and Tamil aksharas", Jl. Universal Computer Science, 2011, Vol. 17, No. 1, pp. 94-106.
- Rakesh R, A G Ramakrishnan, "Fusion of complementary online and offline strategies for recognition of handwritten Kannada characters", Journal of Universal Computer Science, 2011, Vol. 17 (1), pp. 81-93.
- Suresh Sundaram and A G Ramakrishnan, "Attention-feedback based robust segmentation of online Tamil words", under review, Pattern Recognition, 2011.

Improving Tamil-English Cross-Language Information Retrieval by Transliteration Generation and Mining

A Kumaran, K Saravanan & Ragavendra Udupa
Multilingual Systems Research
Microsoft Research India
Bangalore, India.
{kumarana, v-sarak, raghavu}@microsoft.com

Abstract

While state of the art Cross-Language Information Retrieval (CLIR) systems are reasonably accurate and largely robust, they typically make mistakes in handling proper or common nouns. Such terms suffer from compounding of errors during the query translation phase, and during the document retrieval phase. In this paper, we propose two techniques, specifically, transliteration generation and mining, to effectively handle such query terms that may occur in their transliterated form in the target corpus. Transliteration generation approach generates the possible transliteration equivalents for the out of vocabulary (OOV) terms during the query translation phase. The mining approach mines potential transliteration equivalents for the OOV terms, from the first-pass retrieval from the target corpus, for a final retrieval. An implementation of such an integrated system achieved the peak retrieval performance of a MAP of 0.5133 in the monolingual English-English task, and 0.4145 in the Tamil-English task. The Tamil-English cross-language retrieval performance improved from 75% to 81% of the English-English monolingual retrieval performance, underscoring the effectiveness of the integrated CLIR system in enhancing the performance of the CLIR system.

1. Introduction

With the exponential growth of non-English population in the Internet over the last two decades, Cross-Language Information Retrieval (CLIR) has gained importance as a research discipline and as an end-user technology. While the core CLIR system is fairly robust and accurate with sufficient training data, it's handling of proper or common nouns (or, more generally, those query terms that could occur in their transliterated form in the target corpus in cross language environments) is far from desirable in most implementations. In essence, the name translations are not typically part of translation lexicons used for query translations, and hence do not get translated properly in the target language. Note that, from the CLIR point of view, any un-translated word is an out-of-vocabulary word, which typically include words that are literally transliterated into local language words (such as, *computer, corporation,* etc.), specifically in those countries where English is spoken as a second language. This phenomenon is called as *code-mixing*. Such words are not available in the translation lexicon, but are typically part of the local language corpora. Also, given that a name may be spelled differently in the target corpus – particularly for those names that are not native to the target language – the retrieval performance suffers further, as the errors in query translation and spelling variations compound. Given that proper and common nouns form a significant portion of query terms, it is

critical that such query terms are handled effectively in a CLIR system. This is precisely the research theme that we explore in this paper.

Evaluation of Tamil-English cross-language information retrieval systems started first in the Forum for Information Retrieval Evaluation (FIRE) [1], modeled after the highly successful CLEF [2] and NTCIR [3] campaigns. In 2010, FIRE organized several ad hoc monolingual and cross-language retrieval tracks, and we participated in the English monolingual and cross-language Hindi-English and Tamil-English ad hoc retrieval tracks. This paper presents the details of our participation, specifically, in the Tamil-English cross-language tasks and present the performance of our official runs.

2. Cross-language Retrieval System

In this section, we outline the various components of our CLIR system integrated with the two techniques for handling OOV words.

Our monolingual retrieval system is based on the well-known Language Modeling framework to information retrieval. In this framework, the queries as well the documents are viewed as probability distributions. The similarity of a query with a document is measured in terms of the likelihood of the query under the document language model. We refer interested readers to [6, 7] for the retrieval model and the details of this framework. In our CLIR model, the query in a source language is translated into the target language – English – using a probabilistic translation lexicon, learnt from a given parallel corpora of about 50,000 parallel sentences between English and Tamil. Such learnt translation dictionary included ~107 K Tamil words and ~45 K English words. From this dictionary, we used only top 4 translations for every source word, an empirically determined limit to avoid generation of noisy terms in the query translations.

Like any cross-language system that makes use of a translation lexicon, we too faced the problem of out of vocabulary (OOV) query terms. To handle these OOV terms, we used two different techniques, (i) generation of transliteration equivalents, and (ii) mining of transliteration equivalents:

- In Transliteration Generation, the transliterations of the OOV terms in the target language are generated using an automatic Machine Transliteration system, and used for augmenting the query in the target language.
- In Transliteration Mining, the transliteration equivalents of the OOV terms are mined from the top-retrieved documents from the first pass, which are subsequently used in the query for a final retrieval [8].

2.4 Generating Transliterations

We adopted a conditional random fields based approach using purely orthographic features, as a systematic comparison of the various transliteration systems in the NEWS-2009 workshop [9] showed conclusively that orthography based discriminative models performed the best among all competing systems and approaches. In addition, since the Indian languages share many characteristics among them, such as distinct orthographic representation for different variations – aspirated or un-aspirated, voiced or voiceless, etc. – of many consonants, we introduced a word origin detection module (trained with about 3000 hand-classified training set) to identify specifically Indian origin names. All other

names are transliterated through an engine that is trained on non-Indian origin names. Manual verification showed that this method about 97% accurate. We used CRF++, an open source implementation of CRF model, trained on about 15,000 parallel names between English and Tamil. The transliteration engine was trained on a rich feature set (aligned characters in each direction within a distance of 2 and source and target bigrams and trigrams) generated from this character-aligned data.

2.5 Mining Transliteration Equivalents

The mining algorithm issues the translated query minus OOV terms to the target language information retrieval system and mines transliterations of the OOV terms from the top results of the first-pass retrieval. Hence, in the first pass, each query-result pair is viewed as a "comparable" document pair, assuming that the retrieval brought in a reasonably good quality results set based on the translated query without the OOV terms. The mining algorithm hypothesizes a match between an OOV query term and a document term in the "comparable" document pair and employs a transliteration similarity model to decide whether the document term is a transliteration of the query term Transliterations mined in this manner are then used to retranslate the query and issued again, for the final retrieval. The details of transliteration similarity model may be found in [6, 7], and the details of our training are given in [8].

3. Experimental Setup & Results

In this section, we specify all the data used in our experiments, both that were released for the CLIR experiments for FIRE task, and that used for training our CLIR system.

3.1 FIRE Data

The English document collection provided by FIRE was used in all our runs. The English document collection consists of ~124,000 news articles from "The Telegraph India" from 2004-07. All the English documents were stemmed. Totally 50 topics were provided in each of the languages, each topic having a title (T), description (D) and narrative (N), successively expanding the scope of the query. Table 1 shows a typical topic in Tamil, and the TDN components of the topic, for which relevant English documents are to be retrieved from the aforementioned English news corpus. It should be noted that FIRE has also released a set of 50 English (i.e., target language) topics, equivalent to each of the source language topics..

Table 1. A Typical FIRE Topic in Tamil.

Type	**Topic**
Title	குட்கா தயாரிப்பாளர்களுடனான தாதாக்களின் மறைமுகத் தொடர்பு.
Description	கோவா மற்றும் மாணிக்சந்த் குட்கா தயாரிப்பாளர்களுடனான தாவூத் இப்ராஹிமினின் மறைமுகத் தொடர்பு.
Narration	கோவா மற்றும் மாணிக்சந்த் குட்கா தயாரிப்பாளர்களுடன் பிரபல தாதா தாவூத் இப்ராஹிமின் மறைமுகத் தொடர்பு பற்றிய செய்திககள் இந்த ஆவணத்தில் இடம்பெறலாம். தாவூத் இப்ராஹிமின் மற்ற தயாரிப்பாளர்களுடனான செய்திகள் இதில் இடம்பெறத் தேவையில்லை.

Table 2. A Typical FIRE Topic in English.

Type	Topic
Title	Links between Gutkha manufacturers and the underworld.
Description	Links between the Goa and Manikchand Gutkha manufacturing companies and Dawood Ibrahim.
Narration	A relevant document should contain information about the links between the owners of the Manikchand Gutkha and Goa Gutkha companies and Dawood Ibrahim, the gangster. Information about links between Dawood Ibrahim and other companies is not relevant.

3.2 Metrics & Performance Data

The standard measures for evaluating our tasks were used, specifically, Mean Average Precision (MAP) and Precision at top-10 (P@10). As shown in Table 1, each of the 50 topics in Tamil has a title (T), description (D) and narrative (N), successively expanding the scope of the query. We ran our experiments taking progressively each of (title), (title and description), and (title, description and narrative), calibrating the cross-language retrieval performance at each stage, to explore whether expanding the query adds useful information for retrieval or just noise. Table 3 shows the notation used in our description of various configurations to interpret the results presented in Tables 4 and 5.

Table 3. Notations used

T/TD/TDN	Title/ Title and Description / Title, Description and Narration
M	Transliteration Mining
G_D	Transliteration Generation

Tables 4 and 5 show the results of our monolingual as well as cross-language official runs submitted to FIRE 2010 shared task. The format of the run ids in the results table is 'Source-Target-Data-Technique', where 'Data' indicates the data used for topic, and is one of {T, TD, TDN} and 'Technique' indicates the technique and from the set {M, G_D, G_T, M+G_D, M+G_T}. The '+' refers to the combination of more than one approach. The symbols double star (**) and single star (*) indicate statistically significant differences with 95% and 90% confidence respectively according to the paired t-test over the baseline. The best results achieved are highlighted in bold.

4.4 Monolingual English Retrieval

We submitted 3 official runs for the English monolingual track, as shown in the Table 4. For these runs, the English topics provided by the FIRE 2010 organizers were used.

Table 4. English Monolingual Retrieval Performance

Run	MAP	P@10
English-English-T	0.3653	0.344
English-English-TD	0.4571	0.406
English-English-TDN	**0.5133**	**0.462**

With the full topic (TDN), our system achieved a peak MAP score of 0.5133. Generally this performance is thought to be the upper bound for cross-language performance, presented in Table 5.

4.6 Tamil-English Cross-Language Retrieval

We submitted totally 12 official Tamil-English cross-language runs, as shown in Table 5. As discussed for the Hindi-English runs, the first run under each of the 'T', 'TD' and 'TDN' sections in Table 5 present the results of the runs without handling the OOV terms, and hence provide a baseline for measuring the incremental performance due to transliteration generation or mining, provided subsequently.

Table 5. Tamil-English Cross-Language Retrieval Performance

Run	MAP	P@10
Tamil-English-T	0.2710	0.258
Tamil-English-T[G_D]	**0.2891***	**0.268**
Tamil-English-T[M]	0.2815**	0.258
Tamil-English-T[M+G_D]	0.2816*	0.268
Tamil-English-TD	0.3439	0.346
Tamil-English-TD[G_D]	0.3548*	0.35
Tamil-English-TD[M]	**0.3621****	0.346
Tamil-English-TD[M+G_D]	0.3617**	**0.362**
Tamil-English-TDN	0.3912	0.368
Tamil-English-TDN[G_D]	0.4068**	0.378
Tamil-English-TDN[M]	**0.4145****	0.368
Tamil-English-TDN[M+G_D]	0.4139**	**0.394**

From the results presented in Table 5, we observe that the usage of all of the components of the topic, namely T, D and N, produced the best retrieval performance. The basic Tamil-English cross-language run 'Tamil-English-TDN' (without transliteration generation or mining), achieved the MAP score

0.3912, and our best cross-language run 'Tamil-English-TDN[M]' with mining achieved a MAP score of 0.4145. We observe, in general, similar trends in the other runs that use only the title, or title and description sections of the topics. While the cross-language performance of Tamil-English achieves ~81% of our monolingual English retrieval performance, we observe that this is not as high as the Hindi-English retrieval, perhaps due to the highly agglutinative nature of Tamil.

Given that the mining technique performed generally above the other techniques, we focus on addition of mining to the base CLIR, for subsequent analysis in the following sections.

4.7 Mining OOV terms and its effect on CLIR performance

In this section, we analyze the volume of the OOV terms in FIRE topics, and to what extent they are handled by our mining technique, which clearly emerged as the better technique for boosting the retrieval performance. Also, we show the effect of handling the OOVs on the cross-language retrieval performance, for both the Hindi-English and Tamil-English CLIR runs. Table 6 enumerates the number of unique OOV terms in Tamil FIRE 2010 topics.

Table 6. OOV terms in Tamil-English CLIR.

Topic Config.	OOV terms	Transliteratable OOV terms	Transliteratable OOV terms handled	% of Transliteratable OOV terms handled
T	24	13	5	38.46
TD	58	29	15	51.72
TDN	129	47	24	51.06

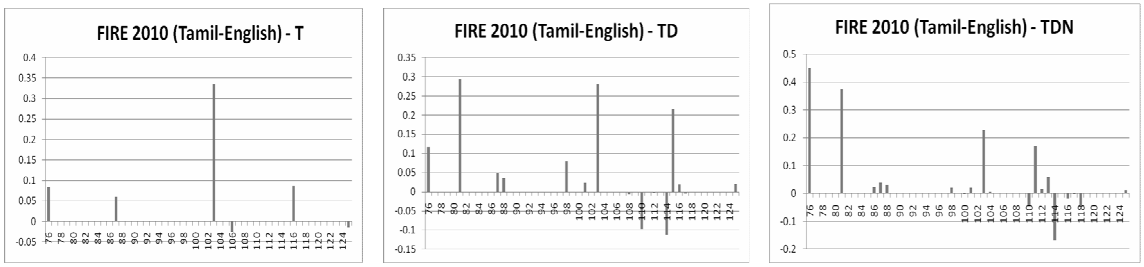
As shown in the third line in Table 6 in the TDN configuration of the 50 Tamil topics of the FIRE2010 shared task, there were totally 129 unique OOV terms, out of which 47 were proper or common nouns, handling of which may help improving the cross-language retrieval performance. Our mining technique did find at least one transliteration equivalent for 24 of these OOV terms (that is, 51%). As shown in Table 5, handling these OOV's resulted in nearly 6% improvement in the MAP score over the baseline where no OOV's were handled.

Note that Tamil OOV's pose specific challenges as outlined below: First, the transliteratable terms mentioned in the third column of the Table 6 excludes some terms whose equivalents are multiword expression in English; mining such multiword transliteration equivalents is beyond the scope of our work and hence they were not handled. Second, 26 out of the 47 terms that are transliteratable were inflected or agglutinated. While our mining algorithm could mine some of them, many of the terms were missed at our parameter settings for mining. By relaxing the mining parameters settings we could mine more such terms, but such relaxation introduced many more noisy terms, affecting the overall retrieval performance. We believe that the use of a good stemmer for inflectional languages like Tamil may help our mining algorithm and, compositionally, the cross-language retrieval performance.

4.8 Mining OOV terms and its effect on individual topic performance

In this section, we discuss effect of our approaches on individual topics of the FIRE 2010 shared task, both for Hindi-English and Tamil-English tasks. Figure 1 shows the difference in the Average Precision – topic-wise – between the baseline CLIR system and that integrated with our mining technique. Individual figures provide the differences for each topic, in each of the three configurations T, TD and TDN, for the Tamil-English tasks. We see that many more topics benefitted from the mining technique; for example, in the Tamil-English language pair, in TDN configurations, 11 topics were improved (with 3 of them with an improvement of ≥ 0.2 in MAP score) whereas only 4 topics were negatively impacted (all of them droppin < 0.2 in MAP score). Similar trends could be seen for all configurations, in the Tamil-English tasks.

Fig. 1. Differences in Average Precision between the baseline and CLIR with mining



From Table 6, we note that the retrieval performance in Tamil-English test collection mining brings maximum improvement of 6% over the baseline in TDN setup.

5. Conclusion

In this paper, we underscored the need for handling proper and common nouns for improving the retrieval performance of cross-language information retrieval systems. We proposed and outlined two techniques for handling out of vocabulary (OOV) words – using transliteration generation, and transliteration equivalents mining – to enhance a state of the art baseline CLIR system. We presented the performance of our system under various topic configurations, specifically for English monolingual task and Tamil-English cross-language tasks, on the standard FIRE 2010 dataset. We show that the performance of our baseline CLIR system is improved significantly by each of the two techniques for handling OOV terms, but consistently more so by the mining technique. Significantly, we also show empirically that the performance of the CLIR system enhanced with transliteration mining is close to that of monolingual performance, validating our techniques for handling OOV terms in the cross-language retrieval.

References

- Forum for Information Retrieval Evaluation. http://www.isical.ac.in/~fire/
- The Cross-Language Evaluation Forum (CLEF). http://clef-campaign.org
- NTCIR: http://research.nii.ac.jp/ntcir/
- Peters, C.: Working Notes for the CLEF 2006 Workshop (2006)

- Majumder, P., Mitra, M., Pal, D., Bandyopadhyay, A., Maiti, S., Mitra, S., Sen, A., Pal, S.: Text collections for FIRE. In: Proceedings of SIGIR (2008)
- Jagarlamudi, J., Kumaran, A.: Cross-Lingual Information Retrieval System for Indian Languages. In: Working Notes for the CLEF 2007 Workshop (2007)
- Udupa, R., Jagarlamudi, J., Saravanan, K.: Microsoft Research India at FIRE2008: Hindi-English Cross-Language Information Retrieval. In: Working notes for Forum for Information Retrieval Evaluation (FIRE) 2008 Workshop (2008)
- Udupa, R., Saravanan, K., Bakalov, A., Bhole, A.: "They Are Out There, If You Know Where to Look": Mining Transliterations of OOV Query Terms for Cross-Language Information Retrieval. In: 31th European Conference on IR Research, ECIR (2009)
- Li, H., Kumaran, A., Pervouchine, V., Zhang, M.: Report of NEWS 2009 Machine Transliteration Shared Task. In: ACL 2009 Workshop on Named Entities (NEWS 2009), Association for Computational Linguistics (2009)
- Kumaran, A, Khapra, M., Bhattacharyya, P: Compositional Machine Transliteration. ACM Transactions on Asian Language Information Processing (TALIP) (2010)