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

உலகத்தமிழ் தகவல் தொழில்நுட்ப மன்றம்
(உத்தமம்)
International Forum for Information Technology in Tamil
(INFITT)

20ஆம் உலகத் தமிழ் இணைய மாநாடு (மெய்நிகர்)-2021

20th International Tamil Internet Conference (Virtual)–2021

December 2-5, 2021

மாநாட்டுக் கட்டுரைகள்
Conference Papers

தொகுப்பு

மா. கணேசன்

கி. பரமேஸ்வரி

பதிப்பு: உலகத்தமிழ் தகவல் தொழில்நுட்ப மன்றம்

Proceedings of the Tamil Internet Conference, 2021 (Virtual)
http://www.uttamam.org/infitt_papers.php

Contents

Tamil News Paper Text Summarization using Generative Adversarial Network

L. Visaka
Dept. of Information Technology
Chennai Institute of Technology
Email: visakalit2019@citchennai.net

R. Anusha
Dept. of Information Technology
Chennai Institute of Technology
Email: anusharit2019@citchennai.net

R. Ramesh
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ramesh@citchennai.net

R. Ponnusamy
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ponnusamyr@citchennai.net

Abstract

Automatic Tamil Text Summarization is one of the hard tasks while processing the Natural Language Processing. In general two methods are existing for text summarization, one is extractive summarization and the other method is abstractive summarization. The main problem with the existing summarizer is that the grammatical readability of the text is questionable. Secondly, most of the existing neural network-based models generate a trivial summary. Therefore, a new alternative approach is required to solve this problem. In this work, an attempt is made to design and develop an abstractive summarizer using a Generative Adversarial Network. The nouns, verbs are extracted and the term frequency, inverse document frequency is taken from the training and testing documents by using python NLTK POS taggers. This input is given to the Generative Adversarial Network machine to generate and discriminate the next word in the sequence given the previous word. Which selects the most essential information from the given system. After that, the final sequence of summarized text is generated which captures the essential information from the original text. At first, the system uses the Tamil daily newspaper Dailythanthi dataset collected from the newspaper data for both training and testing. The standard measures like, precision, recall, and F1-measure are presented.

Keywords: Natural Language Processing, Tami text, Summarization, Generative Adversarial Network, Abstractive Summarization, Precision, Recall, F1-measure.

1. Introduction

Text summarization is one of the essential tasks in many Natural Language Processing, Information Retrieval, and Recommendation Systems. In general, there are two methodologies available for text summarization. The first is extractive summarization and the Second method is abstractive summarization. Extractive summarization gets only the important sentences from the given set of sentences. Alternatively, in the case of abstractive summarization, it generates the set of main text ideas from the given set of input. The latter method is difficult compared to the first one.

There are several methods are available for Tamil text summarization. One of the important methods known as the seq2seq model generally uses the principle of maximum likelihood estimation (MLE) technique. It undergoes revelation bias during the interpretation phase. In this discrepancies between teaching and reasoning occur cumulatively with the order and become more pronounced as the extent of the arrangement increases. Therefore, the consistency and readability of the generated summaries remain inadequate, especially when the seq2seq model is applied directly to long articles.

It is proved that GAN-based summarization system design gives good summarization for the English language. So, a similar thing can be an application for the Tamil language also. Therefore, in this work, an attempt is made to summarize the documents using GAN Network for Tamil language text. The output of the summarization is also good compared to the regular MLE-based methods.

In this paper, Section 2 reviews the literature, Section 3 describes basic system design methods. Section 5 provides an overview experimental setup in section, 6 which explains the result and discussion. Section 7 Conclusion about the interpretation of the paper.

2. Background Literature

The Tamil text summarization has an important tool for different Natural Language Processing and Information Retrieval and Recommendations. A vast amount of work has been carried out to summarization the text worldwide by considering its importance using different algorithms and modern tools. This section deliberates works related to text summarization systems development.

1. Thevatheepan Priyadharsan and Sagara Sumathipala 2019[1] summarized the Tamil text for Tamil online sports news using the NLP method. The text summarization is carried out with six sub-processes and the accuracy F-measure of this text summarization system is 76.6% which is higher than the existing approach for the Tamil text summarization.
2. Rupal Bhargava and et al., in 2020[2] attempted to summarize the text using paraphrase detection. This algorithm is used to reduce text verbosity by removing duplicate paraphrases. This approach is proposed in this article for abstract text summarization, which uses a Generative Adversarial Network to perform multilingual text summarization.

3. Syed Sabir Mohamed and Shanmuga Sundaram Hariharan in 2016[3] attempted to summarize the Tamil text using the centroid approach. The paper follows on generating summaries using a Centroid-based algorithm. The authors reported 82.59% of results in one of their documents.
4. Hao Xu and et al., in 2018[4] proposed a sequence GANs approach for long text summarization using Generator with double-attention, Discriminator with triple RNNs, and Policy gradient for training GAN. The model is still a supervised learning one relying on high-quality training datasets which is scarce.
5. Apurva D.Dhawale and et al., in 2020[5] reviewed the advancing era of text summarization in Indian and foreign languages using the NLP method. Work can be divided into monolingual, bilingual, monolingual, and multilingual summaries.
6. Ms. P. Mahalakshmi and et al., in 2021[6] tried a cross-language Multi-Document summary model using machine learning technology. In determining the summary score associated with the F measurement, the predicted NBC method resulted in a higher F measurement of 91.64%, while the CTLC methodology achieved an average F measurement of 86%.
7. Siddhartha Banerjee and et al., in 2015[7] A MultiDocument abstract summary proposed using ILP-based MultiSentence compression. This includes statement clustering and summary sentence generation. Experimental results from the 2004 and 2005 DUC datasets show that the proposed approach outperforms all baselines and the latest extract summaries.
8. Yenliang Chen and et al., in 2021[8] have developed a template approach for summarizing restaurant reviews using the TextRank algorithm. This way, users can quickly get positive and negative opinions about all the important aspects of the restaurant.

3. System Design

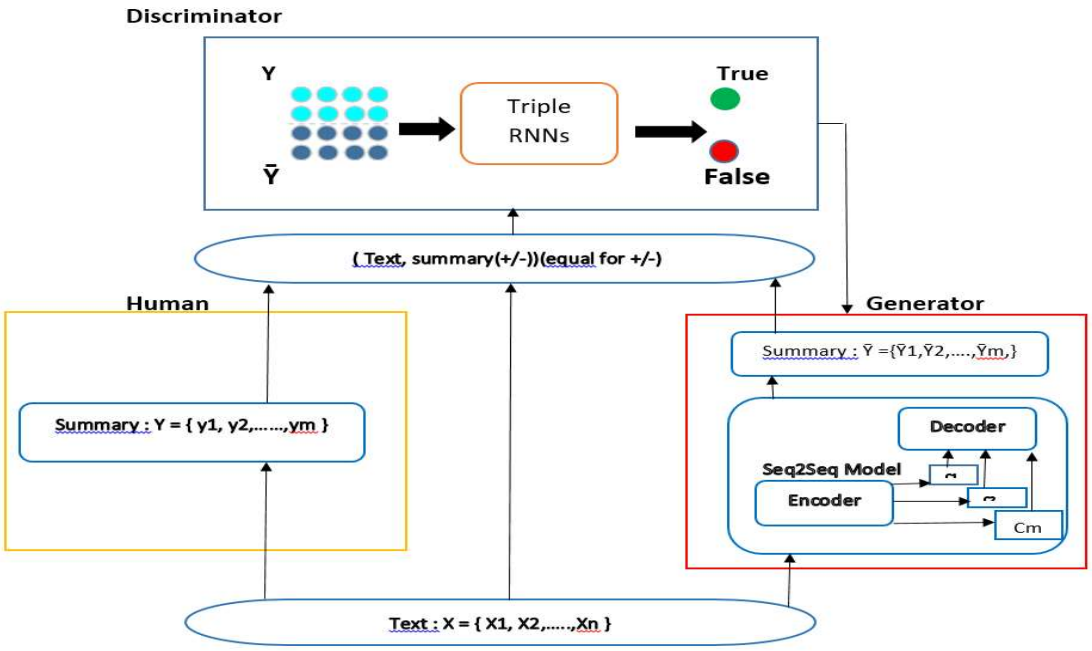


Figure-1 Adversarial Text Summarization Framework

The general text summary block diagram is shown in Figure 1. The input data is a text string of n words X = {x1, x2... xn}, where xi is a word. For the X source, Y = {y1, y2... ym} is called the summary of the underlying truth and Y' = {y1', y2'... Y'm'} is the automatically generated summary.

Generator with Double-Attention

As shown in Figure 2, the generator, abbreviated as G, is a model of an encoder-decoder for sequence conversion. The goal of G is to generate a summary Y′ of a particular text x formalized as Yi′G (Y′1: i 1 | X1: n). Where Y′1: i is one of the steps. Partial summarization means generated and me. To find useful parts of the source code, we introduced the attention mechanism IARNN-WORD in X to solve the attention switching problem. Instead of using plain text words for your encoder, consider presenting the words to your encoder as follows:

$$\alpha_i = \sigma(r_t^T M_{ik} x_i) \qquad (1)$$

$$\tilde{x}_i = \alpha_i * x_i \qquad (2)$$

Where Mik is an attention matrix that turns the representation of the source text rt into a space consisting of words. Here we redefine a conditional probability for G as follows:

$$G(y_i^J | Y_1^J{:}i-1, \tilde{X}) = g(y_i^J - 1, s_i, c_i) \qquad (3)$$

Where si is the hidden state unit in the decoder and ci is the context vector in step i. For the standard GRU decoder, the latent state si is a function of the state of the previous step si - 1, of the output of the previous step y′i - 1, and of the ith context vector:

$$s_i = f(s_{i-1}, y_i^J - 1, c_i) \qquad (4)$$

To generate the ith word of the summary, G uses all of X's input, and the decoder recovers each word to which a unique context vector corresponds. The context vector is defined as follows:

$$c_i = \sum_{J=1}^{n} \beta ij\ hj \qquad (5)$$

The weight β is defined as follows:

$$\beta ij = -\exp(eij) \frac{\exp[e_{ij}]}{\sum_{k=1} \exp[e_{ik}]} \qquad (6)$$

Where eij = a(sij, hi) is called the alignment model, which evaluates the goodness of fit from the jth word of the text and the ith word of the summary. This is the traditional attention mechanism used on the . decoder

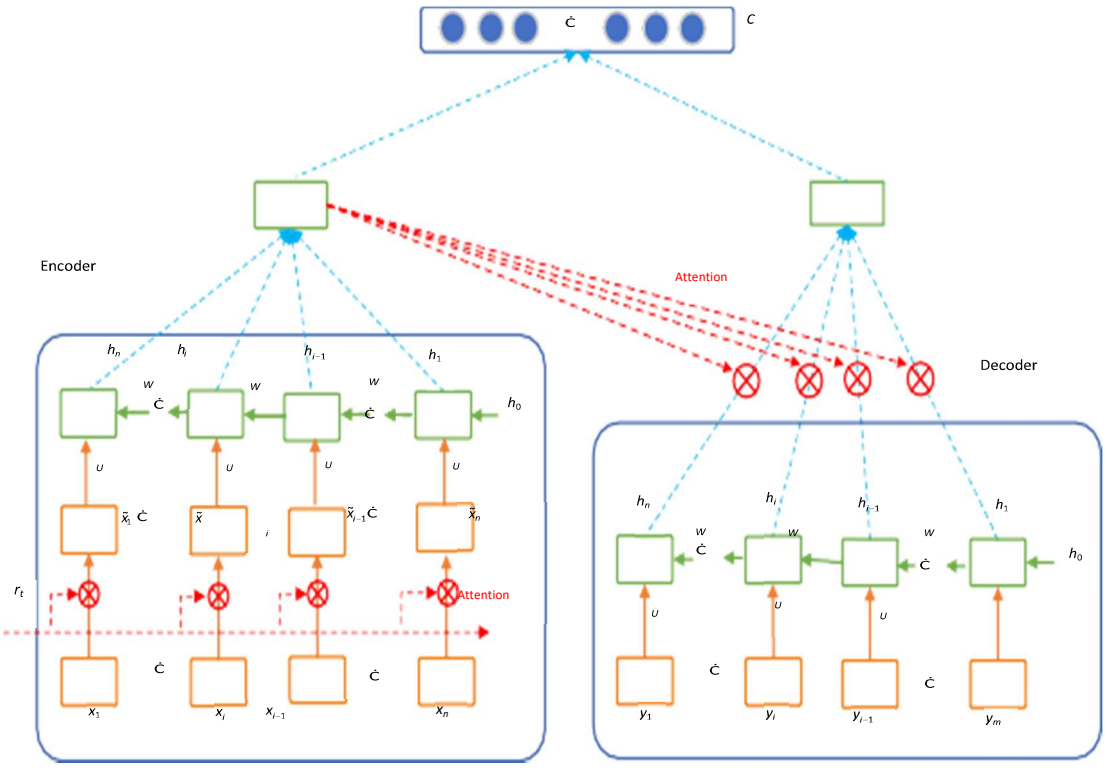


Figure-2 The Seq2Seq Generator Network

Discriminator with Triple RNNs

Distinguishing D is used to distinguish the generated summary from the actual summary as much as possible. This is a typical problem of binary classification. Previous studies have shown that RNNs are suitable for word processing tasks, especially long texts, such as object selection and classification. Therefore, we use RNNs as the basic module of the identifier architecture. Train the discriminator using the mini-batch method to avoid folding mode. Here we use the human message (X, Y) as a positive example and the scanned message (X, Y) as a negative example. Here Y

G (• | X). At the start, we use the same batch size to randomly load (Xi, Yi) and (Xi, Yi) into the discriminator. For each pair of text summaries passed to the RNN, Hcontent is the hidden state vector of the source text and H Résumé is the hidden state vector of the summary. These two RNNs for feature extraction shared parameters. Then, pairs of abstract feature vectors H = [(Hc 0, Hs 0), (Hc 1, Hs 1), ..., (Hc k, Hs k)] are fed into the third, active RNN. as a binary classifier. The last layer is a softmax layer for the probability (X, Y) coming from the underlying truth data, i.e. D(X, Y). The optimization goal of D is to minimize the cross-entropy loss for binary classification. Policy Gradient for Training GAN

The adversarial summarization framework aims to encourage the generator to generate summaries that make it the discriminator difficult to distinguish them from real ones. So, the ultimate goal function of training is:

$$\min_{G}\max_{D} v_{[D,G]} = \mathbf{E}(\mathbf{x},\mathbf{y}) \sim \mathbf{Pr}(\mathbf{X},\mathbf{Y})[\mathbf{logD}(\mathbf{X},\mathbf{Y})] + \mathbf{E}(\mathbf{X},\mathbf{Y}) \sim \mathbf{Pg}(\mathbf{X},\mathbf{Y})[\mathbf{1}\,\mathbf{logD}(\mathbf{X},\mathbf{Y})] \quad (7)$$

Where Pr(X, Y) is a human sampled text pair and Pg(X, Y) is that of the generator, Y` ~ G (• | X). That is, G tries to produce a high-quality summary to fool D, while D tries to distinguish between the generated summary and the underlying truth.

$$\theta^{\pi} best = \arg M_{\theta^{\pi}}^{ax} \sum_{A_i \in \theta^{\pi}}^{\dot{I}} best\ Reward(Si, Yi) \quad (8)$$

RL Strategy can evaluate every possible action in any state given the environmental feedback of the reward and find the action to maximize the expected reward E (i yi∈θπ Part Reward (si, yi), θπ). On this basis, we assume that the generated summary is rewarded with the actual summary by D, denoted R(X, Y). We express the parameters as part of the encoder-decoder by θ, then our objective function is expressed by maximizing the expected reward of the generated summary based on the RL:

$$\begin{aligned} \theta_{best}^{\pi} &= arg\ m_{\theta}^{ax}\ E\left(R(X,Y')\right) \\ &= \arg m_{\theta}^{ax}\ \Sigma_x \Sigma_y, P\theta(X,Y')R(X,Y') \quad (9) \\ &= \arg m_{\theta}^{ax}\ \Sigma_x P(X) \Sigma_y, (Y'|X)R(X,Y') \end{aligned}$$

Where Pθ (X, Y`) represents the probability of some text summaries (X, Y`) under the parameter θ. We redefine the right-hand side of equation (9) to be Jθ, which is the expected reward when G gets the optimal parameter. The probability distribution of each pair of key documents (Xi, Y`i) can be considered as uniformly distributed:

$$\begin{aligned} \mathbf{J}\boldsymbol{\theta} &= \Sigma x\ P(X)\Sigma y, P\theta(\mathbf{Y}'|\mathbf{X})\mathbf{R}(\mathbf{X},\mathbf{Y}') \\ &\approx \frac{1}{n}\sum_{i=1}^{n} R(Xi, Yi') \quad (10) \end{aligned}$$

Whose Gradient w.r.t is:

$$\begin{aligned} \nabla_{j\theta} &= \Sigma_x P(X)\Sigma_y, R(X,Y')\nabla P_{\theta}(Y'|X) \\ &= \Sigma_x P(X)\Sigma_y, R(X,Y')P_{\theta}(Y'|X)\frac{\nabla_{P\theta}[y'|x]}{p_{\theta}(y^1|x)} \\ &= \Sigma_x P(X)\Sigma_y, R(X,Y')\nabla P_{\theta}(Y'|X)\nabla log\ P\theta(Y'|X) \\ &\approx \frac{1}{n}\sum_{i=1}^{n} R(Xi,Yi')\ \nabla log\ P\theta(Y'|X) \quad (11) \end{aligned}$$

The Gradient approximation is used to update $\theta, where\ \alpha\ denotes\ learning - rate$:

$$\theta_{i+1} = \theta^{i} + \alpha \nabla J_{\theta i} \quad (12)$$

As a result, the key to gradient optimization is to calculate the probability of the generated summaries. As the model parameters are updated, the model gradually improves summarization and reduces losses. Expected rewards can be estimated by sampling. In the training process,

weakening on one side leads to a break in the fight, the problem of fashion collapse. So I took over the Monte Carlo search, recorded a partial decode, calculated the average of all possible rewards, and added it to the subsequent sequence. More precisely, if t = n, the decrypting result is only a partial result and the reward is R (Xi, Y`1: i). Based on the gradient of the policy, the discriminator on the other side tells the generator to generate a summary similar to the ground truth. Ideally, the dispersal of the produced summaries and the distribution of the actual summaries completely overlap.

4. Experimental Setup

The dataset used for Summarization is received from the Dailythanthi free open website. It is a Dailythanthi News Dataset having 50 News Articles of fifteen different categories. This dataset includes four different attributes, such as News Id, News summary, News Category, News Title, and News. In this dataset the given dataset is given to the experimentation system and a summarized abstract is received along with the title.

4.1 Experiment Setup

A python program is developed with the TensorFlow, Numpy, Scipy, Nltk, Cuda packages to model the GAN model (TextGAN) to generate text summaries. This prototype works well for state-of-the-art data in the database. In this program, it will be able to expect only the GAN model alone. The system is configured to run in a standalone PC Environment.

The system is simulated and testing and the sample test results are presented as follows

தடுப்பூசி நிலவரம்; மாவட்ட நீதிபதிகளுடன் பிரதமர் மோடி ஆய்வு..!
பதிவு: அக்டோபர் 31, 2021 15:35 PM
புது டெல்லி,

கொரோனா தடுப்பூசிகள் குறைவான அளவில் செலுத்தப்பட்டுள்ள மாவட்டங்களில் பிரதமர் மோடி ஆய்வு நடத்த உள்ளார்.ஜி20 மாநாடு முடிந்து திரும்பியதும், தடுப்பூசி சதவீதம் குறைவாக செலுத்தப்பட்டுள்ள மாவட்டங்களின், மாவட்ட தலைமை நீதிபதிகளுடன் வீடியோ கான்பரன்சிங் மூலமாக நடைபெற உள்ள ஆய்வுக் கூட்டத்தில் மோடி பங்கேற்கிறார்.

அந்த கூட்டத்தில் தடுப்பூசி செலுத்தும் விகிதத்தை அதிகரிப்பதற்கான நடவடிக்கைகள் குறித்து விவாதிக்கப்படும் என்று தெரிகிறது. நவம்பர் 3ம் தேதி மதியம் 12 மணிக்கு இந்த கூட்டம் நடைபெற உள்ளது.

50% முதல் தவணை தடுப்பூசி மற்றும் குறைந்த அளவிளான இரண்டாம் தவணை தடுப்பூசி செலுத்தப்பட்டுள்ள மாவட்டங்கள் இந்த கூட்டத்தில் கலந்து கொள்கின்றன.

அதன்படி, ஜார்க்கண்ட், மணிப்பூர், நாகாலாந்து, அருணாசலபிரதேசம், மராட்டியம், மேகாலயா மற்றும் பிற மாநிலங்களில் உள்ள 40க்கும் மேற்பட்ட மாவட்டங்கள் இந்த கூட்டத்தில் கலந்து கொள்கின்றன.மேற்கண்ட மாநில முதல்வர்களும் இந்த கூட்டத்தில் கலந்து கொள்கின்றனர்.

இதுவரை இந்தியாவில் 106 கோடிக்கும் அதிகமான கொரோனா தடுப்பூசிகள் செலுத்தப்பட்டுள்ளன.கடந்த 24 மணி நேரத்தில் மட்டும் 68 லட்சத்து 4 ஆயிரத்து 806 தடுப்பூசிகள் செலுத்தப்பட்டுள்ளன.
தடுப்பூசி நிலவரம்; மாவட்ட நீதிபதிகளுடன் பிரதமர் மோடி ஆய்வு..! 50% முதல் தவணை தடுப்பூசி மற்றும் குறைந்த அளவிளான இரண்டாம் தவணை தடுப்பூசி செலுத்தப்பட்டுள்ள மாவட்டங்கள் இந்த கூட்டத்தில் கலந்து கொள்கின்றன.

5. Results and Discussion

The performance of the Tamil text summarization arrangement can be appraised by using four different standard metrics is Precision, Recall, Accuracy, and F1 measure. Precision dealings the factualness of the summarizer system. Higher precision means fewer false positives, while a lower precision means false positives.

$$\text{Precision} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of extracted text}}$$

Recall measures the completeness, or sensitivity, of a classifier. Higher recall means fewer false negatives, while lower recall means more false negatives.

$$\text{Precision} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of annotated text}}$$

The correct classification instance measured by,

$$\text{Accuracy} = \frac{Number\ of\ correctly\ classified\ instances}{\text{Total Number of instances}}$$

A harmonic F1-measure is finding the mean of recall and precision

$$\text{F1} - \text{Mesure} = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 1 Performance of Results Evaluation of GAN Summarizer Process

Precision	Recall	F1-Score	Accuracy
92	90	90	91

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) was used as the automatic evaluation method. The ROUGE dimension family, based on n-gram similarity, was first

introduced in 2003. Assume a set of reference summaries-reference summary sets (RSS) created by annotators. The ROUGE-n score for the candidate summary is calculated as follows:

ROUGE-1	0.90

6. Conclusion

Tamil Text Summarization is one of the essential tasks in Text Information Retrieval and Natural Language Processing. Developing and acquiring more accurate inferences for Text Summarization is a challenging task. This paper proposes a GAN model for Tamil Text summarization. In this algorithm, the summarization gives an accuracy is 94 percent which is comparatively good as compared to the traditional summarizers.

7. References

1. Priyadharshan, T., & Sumathipala, S. (2018). Text summarization for Tamil online sports news using NLP. 2018 3rd International Conference on Information Technology Research (ICITR). https://doi.org/10.1109/icitr.2018.8736154.
2. Bhargava, R., Sharma, G., & Sharma, Y. (2020). Deep text summarization using generative adversarial networks in Indian languages. Procedia Computer Science, 167, 147-153. https://doi.org/10.1016/j.procs.2020.03.192.
3. Mohamed, S. S., & Hariharan, S. (2016). A Summarizer for Tamil language using centroid approach. International Journal of Information Retrieval Research, 6(1), 1-15. https://doi.org/10.4018/ijirr.2016010101.
4. Dhawale, A. D., Kulkarni, S. B., & Kumbhakarna, V. (2020). Survey of progressive era of text summarization for Indian and foreign languages using natural language processing. Innovative Data Communication Technologies and Application, 654-662. https://doi.org/10.1007/978-3-030-38040-3_74.
5. Yang, W., Hua, R., & Zhao, Q. (2019). Sequence generative adversarial network for Chinese social media text summarization. 2019 Chinese Automation Congress (CAC). https://doi.org/10.1109/cac48633.2019.8996751.
6. Et.al, M. P. (2021). Cross - Language based multi-document summarization model using machine learning technique. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 12(6), 331-335. https://doi.org/10.17762/turcomat.v12i6.1393.
7. Suleiman, D., & Awajan, A. A. (2019). Deep learning based extractive text summarization: Approaches, datasets and evaluation measures. 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS). https://doi.org/10.1109/snams.2019.8931813.
8. Jin, H., & Wan, X. (2020). Abstractive multi-document summarization via joint learning with single-document summarization. Findings of the Association for Computational Linguistics: EMNLP 2020. https://doi.org/10.18653/v1/2020.findings-emnlp.231.

Multi-Source Tamil Text Summarization using Latent Semantic Analysis (LSA)

R. Vaishnavi
Dept. of Information Technology
Chennai Institute of Technology
Email: rvaishnaviit2019@citchennai.net

Nandhini G. K
Dept. of Information Technology
Chennai Institute of Technology
Email:nandhinigkit2019@citchennai.net

R.Saravanan
Department of Information Technology
Chennai Institute of Technology
Email:saravananr@citchennai.net

R. Ponnusamy
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ponnusamyr@citchennai.net

Abstract

Multisource Automatic Tamil Text Synthesis is one of the complex tasks in processing natural language processing. In specific, abstractive summarization is hard to perform as compared to extractive summarization. The main problem with the existing contents is that the grammatical readability of the text is questionable. Secondly, most of the existing neural network-based models generate a trivial summary. Therefore, a new alternative approach is necessary to resolve this issue. In this work, an attempt is made to design and develop an abstractive summarizer using Latent Semantic Analysis. The nouns, verbs are extracted and the term-frequency, inverse document frequency are taken from the training and testing documents by eliminating the stop words. This input is given to the Latent Semantic Analysis machine to extract important features. This is done by multiple text sources and the redundancy elimination is done. Once it is done again these topics are combined and a final summary is generated. At first, the system uses the different daily newspapers on the same topic collected from the newspaper data for both training and testing. Standard measurements like, accuracy, recall, and F1-measure are calculated.

Keywords: Natural Language Processing, Tami Text, Summarization, Latent Semantic Analysis, Abstractive Summarization, Extractive summarization, Precision, Recall, F1-measure.

1. Introduction

Text summarization is one of the essential tasks in many Natural Language Processing, Information Retrieval, and Recommendation Systems. In general, there are two methodologies available for text summarization. The first is extractive summarization and the Second method is abstractive summarization. Extractive summarization gets only the important sentences from the given set of sentences. Alternatively, in the case of abstractive summarization, it generates the set of main text ideas from the given set of input. The latter method is difficult compared to the first one.

There are several methods are available for Tamil text summarization. One of the important methods known as the seq2seq model generally uses the principle of maximum likelihood estimation (MLE) technique. It undergoes revelation bias during the interpretation phase. In this discrepancies between teaching and reasoning occur cumulatively with the order and become more pronounced as the extent of the arrangement increases. Therefore, the consistency and readability of the generated summaries remain inadequate, especially when the seq2seq model is applied directly to long articles.

In this work, an attempt is made to summarize the documents using LSA for Tamil language text. The output of the summarization is also better compared to the regular MLE-based methods. The main objective of this is to summarize the the documents from multiple-sources. However, due certain practical reasons, limited dataset is taken for experimentation. It is collection of articles from the web of different categories.

In this paper, Section 2 reviews the literature, Section 3 describes basic system design methods. Section 4 provides an overview experimental setup in section, 5 which explains the result and discussion. Section 6 Conclusion about the interpretation of the paper.

2. Background Literature

The Tamil text summarization has an important tool for different Natural Language Processing and Information Retrieval and Recommendations. A vast amount of work has been carried out to summarization the text worldwide by considering its importance using different algorithms and modern tools. This section deliberates works related to text summarization systems development.

1. Thevatheepan Priyadharsan and Sagara Sumathipala 2019[1] summarized the Tamil text for Tamil online sports news using the NLP method. The text summarization is carried out with six sub-processes and the accuracy F-measure of this text summarization system is 76.6% which is higher than the existing approach for the Tamil text summarization.

2. Rupal Bhargava and et al., in 2020[2] attempted to summarize the text using paraphrase detection. This algorithm is used to reduce text verbosity by removing duplicate paraphrases. This approach is proposed in this article for abstract text summarization, which uses a Generative Adversarial Network to perform multilingual text summarization.
3. Syed Sabir Mohamed and Shanmuga Sundaram Hariharan in 2016[3] attempted to summarize the Tamil text using the centroid approach. The paper follows on generating summaries using a Centroid-based algorithm. The authors reported 82.59% of results in one of their documents.
4. Hao Xu and et al., in 2018[4] proposed a sequence GANs approach for long text summarization using Generator with double-attention, Discriminator with triple RNNs, and Policy gradient for training GAN. The model is still a supervised learning one relying on high-quality training datasets which is scarce.
5. Apurva D.Dhawale and et al., in 2020[5] reviewed the advancing era of text summarization in Indian and foreign languages using the NLP method. Work can be divided into monolingual, bilingual, monolingual, and multilingual summaries.
6. Ms. P. Mahalakshmi and et al., in 2021[6] tried a cross-language Multi-Document summary model using machine learning technology. In determining the summary score associated with the F measurement, the predicted NBC method resulted in a higher F measurement of 91.64%, while the CTLC methodology achieved an average F measurement of 86%.
7. Siddhartha Banerjee and et al., in 2015[7] A MultiDocument abstract summary proposed using ILP-based MultiSentence compression. This includes statement clustering and summary sentence generation. Experimental results from the 2004 and 2005 DUC datasets show that the proposed approach outperforms all baselines and the latest extract summaries.
8. Yenliang Chen and et al., in 2021[8] have developed a template approach for summarizing restaurant reviews using the TextRank algorithm. This way, users can quickly get positive and negative opinions about all the important aspects of the restaurant.

3. System Design

In this design, the system is having documents, and terms are used for summarizing the document. Initially, the Tamil text documents are collected from the articles of different newspapers and a matrix is for with row-wise terms and column-wise documents. Matrix X with (i,j) describes with the ith term with jth document representing the frequency of the term in the document.

$$\mathbf{t}_i^T \rightarrow \begin{bmatrix} & & \mathbf{d}_j & & \\ & & \downarrow & & \\ x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,j} & \cdots & x_{m,n} \end{bmatrix}$$

The row matrix relations of different documents for the particular term.

$$\mathbf{t}_i^T = [\, x_{i,1} \quad \cdots \quad x_{i,j} \quad \cdots \quad x_{i,n} \,]$$

In the someway, the column matrix represents how one document represents the different terms.

$$\mathbf{d}_j = \begin{bmatrix} x_{1,j} \\ \vdots \\ x_{i,j} \\ \vdots \\ x_{m,j} \end{bmatrix}$$

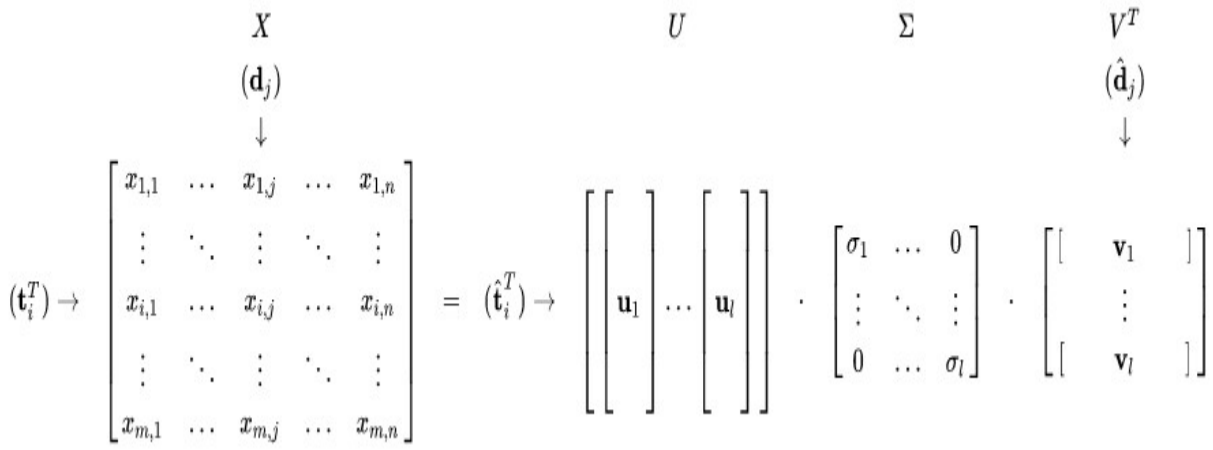
The dot product represents the correlation between two terms. The matrix product $\mathbf{XX^T}$ contains all these dot products and hence giving their correlation. Now, we perform the singular value decomposition by decomposing the matrix X into U and V orthogonal matrices. This is called Singular Matrix decomposition:

$$X = U\Sigma V^T$$

The matrix products giving us the term and document correlations then become:

$$\begin{aligned} XX^T &= (U\Sigma V^T)(U\Sigma V^T)^T = (U\Sigma V^T)(V^{T^T}\Sigma^T U^T) = U\Sigma V^T V\Sigma^T U^T = U\Sigma\Sigma^T U^T \\ X^T X &= (U\Sigma V^T)^T(U\Sigma V^T) = (V^{T^T}\Sigma^T U^T)(U\Sigma V^T) = V\Sigma^T U^T U\Sigma V^T = V\Sigma^T\Sigma V^T \end{aligned}$$

Both products have the same non-zero eigenvalues, now the decomposition looks like this:



The sigma value is called the singular value, and $\mathbf{u}_i$ and $\mathbf{v}_i$ are called the left and right singular vectors. Matrix V^T shows the strength of each word in a sentence or document. The use of this matrix becomes apparent as the implementation progresses. The document vector $\mathbf{d}_j$ is an approximation in low dimensional space. This approximation is described as follows.

$$X_k = U_k \Sigma_k V_k^T$$

Algorithm

The algorithm for LSA consists of three major steps:

i. **Input matrix creation**: By creating the matrix X as matrix representing the document and term frequency as given in the above matrix.

ii. **Make Tf-IDF (Term Frequency-Inverse Document Frequency)**: the cell is filled in with the tf-idf value of the word. A higher tf-IDF value means that the word is more frequent in the sentence but less frequent in the whole document. The higher value indicates that the word is much more representative for that sentence than others.

iii. **Singular Value decomposition (SVD)**: In this step, SVD is performed. It is representing the Euclidean Vector representing the relationships between words and documents/sentences. It is having the capacity to reduce errors and improve accuracy.

iv. **Sentence Selection**: Using the results of SVD different algorithms are used to select important sentences/words. Here we have used the Topic method to extract concepts and sub-concepts from the SVD calculations and are called topics of the input document. These topics can be sub-topics, and then the sentences are collected from the main topics.

v. **Generate Sentence** : After considering these words/topics in to account a essential generator (Random Generator) used to generate the sentence by taking the words into account.

4. Experimental Design

The dataset used for Summarization is received from the different Tamil News Papers of a free open website. It is a having 120 News Articles of different categories. This dataset includes four different attributes, such as News Id, News summary, News Category, News Title, and News. In this dataset, the given dataset is given to the experimentation system and a summarized abstract is received along with the title. As it is an unsupervised data structure there is no need for training.

4. 1 Experimental Setup

A python program is developed with the Numpy, Sklearn, Nltk packages to model the LSA model to generate text summaries. This prototype works well for state-of-the-art data in the database. In this program, it will be able to expect only the LSA model alone. The system is configured to run in a standalone PC Environment.

5. Results and Discussion

The overall performance of the Tamil textual content summarization association may be appraised via way of means of the usage of 4 distinctive trendy metrics is Precision, Recall, Accuracy, and F1 measure. Precision dealings the factualness of the summarizer system. Higher precision way fewer fake positives, whilst a decreased precision way fake positives.

$$\text{Precision} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of extracted text}}$$

Recall measures the completeness, or sensitivity, of a classifier. Higher take into account manner fewer fake negatives, even as decrease take into account manner extra fake negatives.

$$\text{Precision} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of annotated text}}$$

The correct classification instance measured by,

$$\text{Accuracy} = \frac{Number\ of\ correctly\ classified\ instances}{\text{Total Number of instances}}$$

A harmonic F1-measure is finding the mean of recall and precision

$$\text{F1} - \text{Mesure} = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 1 Performance of Results Evaluation of LSA Summarizer Process

Precision	Recall	F1-Score	Accuracy
89	91	90	89

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) turned into used because of the automated assessment method. The ROUGE measurement family, primarily based totally on n-gram similarity, turned into first brought in 2003. Assume a hard and fast reference summaries-reference precise sets (RSS) created with the aid of using annotators. The ROUGE-n rating for the candidate precise is calculated as follows:

ROUGE-1	0.89

6. Conclusion

Tamil Text Summarization is one of the essential tasks in Text Mining and Recommendation systems. As it is a complex syntax and semantics it is hard to put into a structure and therefore there are too many subjective inferences. This paper proposes an LSA model for Tamil Text summarization. In this algorithm, the summarization gives an accuracy is 89 percent which is comparatively better accuracy as compared to the traditional summarizers.

7. References

1. Priyadharshan, T., & Sumathipala, S. (2018). Text summarization for Tamil online sports news using NLP. 2018 3rd International Conference on Information Technology Research (ICITR). https://doi.org/10.1109/icitr.2018.8736154.
2. Bhargava, R., Sharma, G., & Sharma, Y. (2020). Deep text summarization using generative adversarial networks in Indian languages. Procedia Computer Science, 167, 147-153. https://doi.org/10.1016/j.procs.2020.03.192.
3. Mohamed, S. S., & Hariharan, S. (2016). A Summarizer for Tamil language using centroid approach. International Journal of Information Retrieval Research, 6(1), 1-15. https://doi.org/10.4018/ijirr.2016010101.
4. Dhawale, A. D., Kulkarni, S. B., & Kumbhakarna, V. (2020). Survey of progressive era of text summarization for Indian and foreign languages using natural language processing. Innovative Data Communication Technologies and Application, 654-662. https://doi.org/10.1007/978-3-030-38040-3_74.
5. Yang, W., Hua, R., & Zhao, Q. (2019). Sequence generative adversarial network for Chinese social media text summarization. 2019 Chinese Automation Congress (CAC). https://doi.org/10.1109/cac48633.2019.8996751.
6. Et.al, M. P. (2021). Cross - Language based multi-document summarization model using machine learning technique. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 12(6), 331-335. https://doi.org/10.17762/turcomat.v12i6.1393.

7. Suleiman, D., & Awajan, A. A. (2019). Deep learning based extractive text summarization: Approaches, datasets and evaluation measures. 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS). https://doi.org/10.1109/snams.2019.8931813.
8. Jin, H., & Wan, X. (2020). Abstractive multi-document summarization via joint learning with single-document summarization. Findings of the Association for Computational Linguistics: EMNLP 2020. https://doi.org/10.18653/v1/2020.findings-emnlp.231.

சொல்லோடை: கற்கும்-கருவிகளுக்கு ஒரு சொற்றொடர் படையல்

மு. செல்வக்குமார், ப. தமிழ் அரசன், ச. மலைக்கண்ணன்

Selvakumar Murugan, Tamil Arasan Bakthavatchalam, Malaikannan Sankarasubbu

{Selvakumar.murugan, tamil.arasan, malaikannan.sankarasubbu}@saama.com

Saama AI Research Lab, Chennai

Nov 02, 2021

சாரம்*:* செய்யறிவின் *(Artificial Intelligence)* உறுப்பான கற்குங்கருவியியல் *(Machine Learning)* கருவிகள் கடந்த பத்தாண்டுகளில் எதிர்பாராத அளவிற்கு நுட்பமான செயல்களைப் புரிந்து வருகிறது. படத்தைப் பார்த்து அதில் இருப்பவற்றைக் கண்டறிவது தொடங்கி மொழிபெயர்ப்பு வரை பலதரப்பட்ட சிக்கலான வேலைகளைச் செம்மையுறச் செய்து காட்டியுள்ளது . எனினும் ஐரோப்பிய மொழிகளில் இருக்கும் அளவிற்குத் தமிழ் முதலிய மொழிகளில் அவை திறம்பட இயங்குவதில்லை . இதற்குப் பல கரணிகள் இருப்பினும் , முதன்மையானதாக நாங்கள் கருதுவது போதுமான அளவிற்கு தரவுக்கணங்களும், கற்குங்கருவியியலாளர்களும் இல்லை என்பதே . செம்மையான தரவுக்கணங்களை உருவாக்குவது தரவைச் சேகரிப்பதும் *(collection),* செம்மைப்படுத்துவதும் *(cleaning),* அடையாளமிடுவதும்/சிட்டையிடுதலும் *(annotation/tagging)* முதலிய பல வேலைகளை அடக்கியது. அவற்றை ஒன்றன்பின் ஒன்றாகவும் சில நேரம் இணைத்தும் செய்ய வேண்டியுள்ளது . இது மிகுந்த பணிச்சுமையும், பணச்செலவும் பொதிந்த பணி . தரவுக்கணங்கள் இல்லாமல் கருவிகளுக்குக் கற்றுத்தருவது எளிதல்ல . இப்பணியில் செய்தித் தாள்கள் , தமிழ் விக்கிப்பீடியா , வலைப்பூக்கள் முதலிய எண்ணற்ற வலைத் தளங்களில் இருந்து சேகரிக்கப்பட்ட பனுவல்களைத் தொகுத்து , பிழை களைந்து, நகல் நீக்கி எடுத்தவுடன் முடுக்கிவிடும் நிலையில் சொல்லோடை -2021 தரவுக்கணத்தை பகிர்கிறோம். சேகரித்த பனுவல்களைத் தொடர்களாக பிரித்து, பிறமொழிச் சொற்கள் நீக்கி இதுவரை இல்லாத மிகப்பெரிய சொற்றொடர் தொகுப்பைப் படைத்திருக்கிறோம்.. கூடவே பயிற்றுவிக்கப்பட்ட கருவியின் உட்பொதியையும் வெளியிடுகிறோம்.

Abstract*:* Machine Learning, a subfield of Artificial Intelligence, has shown a substantial impact on every field of technology. It has shown capabilities of solving complex tasks such as image recognition to language translation. However when compared to European languages, the machine learning models perform very badly at low resource languages like Tamil. There are many factors that contribute to this, and we consider two specific factors, scarcity of Tamil NLP researchers and more importantly lack of datasets are the key issues. Creating annotated datasets involves many tedious tasks such as data collection, cleaning, identification/tagging,

and so on, that are burdensome and expensive. Modelling, training a machine learning model is close to impossible without datasets. In this work, we compile news collected from numerous resources such as newspaper websites, Tamil Wikipedia, blogs, etc., and release the dataset in ready to use form. We processed the text and created a colossal phrase collection that we call Cholloadai-2021. This is the largest collection of Tamil sentences to the best of our knowledge. We release this dataset along with a trained model.

Keywords: Machine learning, Artificial Intelligence, Language Modelling, Tamil Corpus, Language Technology

1. **முன்னுரை**

தமிழில் கற்குங்கருவியியலைப் பற்றி நாங்கள் அறிந்த வரை இதுவே முதல் கட்டுரை என்பதால் முடிந்தளவு எல்லாக் கருத்துக்களையும் எளிமையாக விளக்க முயன்றுள்ளோம் . கற்குங்கருவியியல் பயனைப் பற்றியும் , வீச்சைப் பற்றியும் விவரிக்கிறோம் . கற்குங்கருவியியலை எப்படி தமிழ்ப் பனுவல்கள் மேல் ஏவுவது , அதற்கு என்ன தேவை என்பன பற்றியும் விளக்குகிறோம் . தரவுகள் எவ்வளவு முக்கியமானது , அவற்றை எப்படிச் சேகரிப்பது , செம்மை செய்வது என்பன பற்றியும் அலசுகிறோம்.

1.1. **தமிழ் மொழி**

திராவிட மொழிக் குடும்பத்தின் மூத்த மொழிகளில் ஒன்றான தமிழ் ஒரு உயர்தனிச்செம்மொழி. இரண்டாயிரம் ஆண்டுகளில் கல்வெட்டுகளில் , ஓலைச்சுவடிகளில் என பல்வேறு எழுத்து வடிவங்கள் பெற்றாலும் இலக்கணமும் சொற்களஞ்சியம் அழியாமல் தொடர்ச்சியாக பேணப்பட்டிருக்கிறது. கல்வெட்டு, சுவடிகள், தாள், கணினி என தமிழ் கடந்து வந்த பாதை மிக நீண்டது . காலதிற்கேற்ப தன்னை தகவமைத்துக் கொண்ட தமிழ் தற்பொழுது செய்யறிவிலும் தடம் பதிக்க வேண்டும் . அண்மைகாலத்தில் செய்யறிவுத் துறையின் உறுப்பான கற்குங்கருவியியல் காட்சியறிவிலும், மொழியறிவிலும் நிகழ்த்திவரும் முன்னேற்றங்கள் தவிர்க்க முடியாதவை. எனவே மாந்த மொழியாக மட்டும் இருந்து வந்த தமிழ் எந்திரங்களும் பேசும் மொழியாகவும் வடிவெடுக்க வேண்டும் . இதன் பொருட்டு எடுத்த முயற்சியின் விளைவாக எழுந்த பணியே இது.

தமிழில் அடிச்சொல்லோடு காலம் , எண்ணிக்கை, வேற்றுமை முதலியற்றை விளக்க ஒன்று அல்லது பல பின்னொட்டுகள் சேர்ந்து புதுச்சொற்கள் பிறப்பதால் தமிழ் ஒரு ஒட்டுநிலை மொழி . ஒற்றை ஆய்தமும், 12 உயிர்களும், 18 மெய்களும், மெய்களில் உயிரேறி 216 உயிர்மெய் என மொத்தம் 247 அரிச்சுவடி எழுத்துக்களைக் கொண்டது தமிழ் . இருசொற்கள் இணையும் போது எப்படி இணைய

வேண்டும் என்று புணர்ச்சியிலக்கண விதிகள் உண்டு. இப்படிப் பல பரிமானங்களைக் கொண்ட தமிழ் மொழியைக் கணினிக்குக் கற்றுத்தருவது எளிதான காரியமன்று.

1.2. **இயல் மொழி ஆள்கை**

இயன்மொழியாள்கை (natural language processing) என்பது கணினியை மொழியோடு ஊறவாடச்செய்தல். இப்படிச் செய்வதின் பயன்கள் எண்ணற்றவை . கைப்பேசியோ மடிக்கணியோ நாம் பொத்தான்களைத் தட்டி இயக்காமல் , வாய்விட்டுச் சொல்வதைக் கேட்டுச் சொன்னபடி செயல்படுமாயின் கணினியை அனைவரும் பயன்படுத்த வழிசெய்யும் . இயன்மொழியைப் புரிந்த கணினி, கேட்கும் கேள்விகளைப் புரிந்து கொண்டு இணையத்தில் இருக்கும் அறிவுக்களஞ்சியங்களை அணுகிப் படித்து கேள்விக்கேற்ப பதிலைச் சொல்ல வல்லதாகும் ; பலமொழிகளிலும் இருக்கும் இலக்கியங்களையும் , நூல்களையும், சான்றோர் உரைகளையும் மொழிபெயர்த்து தரும்; எழுத்துப் பிழைகளை, இலக்கணப் பிழைகளைக் களையக் கைகொடுக்கும் . இப்படிப் பல நேரடிப் பயன்கள் இருப்பினும் , மாந்த மூளையின்/அறிவின் இயல்புகளை நேரடியாக கண்டறிய முடியாவிடினும், அறிவுடைய எச்சங்கள் மொழியுள் புதைந்து கிடக்கின்றன . கணினியையும், மொழியையும் கூத்தாட வைத்து அவ்வியல்புகளைக் கண்டறிவது மாந்த இனத்தின் அறிவுப்பசிக்குச் சிறந்த விருந்து.

எனினும் கணினிக்கு இயல்மொழியைக் கற்றுத்தருவது எளிமையான வேலையல்ல . இயல்பாய் முளைத்துக் கிளைத்த மாந்த மொழிகள் மிகச் சிக்கலானவை. பேச்சிலும் (எழுத்திலும் கூட) பல விவரங்களைச் சொல்லாமலேயே புரிந்து கொள்ளும் அறிவு மாந்தர்க்கு இருப்பதால் , நமக்கு எந்தச்சிக்கலும் இல்லை. ஆனால் இன்னின்னது இன்னின்ன என்று தெரியாத , உலகைப்பற்றி எந்த அறிவும் இலாத, எண்களிலேயே இயங்கும் கணினியை மொழியோடு உறவாட வைப்பது மிகக்கடிது . அதனால் தான் கணினிகளைக் கையாள ஏதுவாக நிரல்மொழிகள் இருக்கின்றன . இயல் மொழிகளைக் காட்டிலும் இலக்கணத்தில் , சொற்களில், பயன்பாட்டு வகையில் நிரல்மொழிகள் மிகச்சிறியவை எளியவை. நூற்றுக்கும் மேற்பட்ட நிரல்மொழிகள் இருப்பினும், ஒரு நிரல்மொழியில் மொத்த முதற்சொற்களையும் (keywords), இலக்கணத்தையும், இடஞ்சாரா இலக்கண (context free grammar) வடிவில் ஓரிரு தாள்களில் அடக்கிவிட முடியும் . ஒவ்வொரு நிரல்மொழியும் ஒருவராலோ , சிலராலோ ஒரு விதப்பான (specific) பணிகளைக் கணினிக்கு அறிவுறுத்தப் படைக்கப் பட்டவை. இயல் மொழிகளோ அப்படி குறிப்பிட்ட ஒரு சிலரால் வடிவமைக்கப் பட்டவை அல்ல . காலப்போக்கில் புவியமைப்பு, பண்பாடு, வணிகம், அரசியல் முதலிய எண்ணற்ற காரணிகளால் மாற்றங்களை உள்வாங்கியவை. அம்மாற்றங்கள் எழுத்திலும் , ஒலிப்பிலும், சொற்களிலும் நடந்துள்ளன . இலக்கணம் கூட அதற்கு விதிவிலக்கல்ல.

இப்படி மிகச்சிக்கான இயன்மொழியைக் கணினிக்குக் கற்றுத்தர மொழித் தரவுகள்

இன்றியமையாதவை. மொழித்தரவுகள் என்றால் பனுவல்களும் , அவற்றின் மீது ஏற்றப்பட்ட மேந்தரவுகளுமாகும். மேந்தரவுகள் என்பது , பனுவலில் சொல்லப்படும் கருத்தைப்பற்றி விவரங்களாக இருக்கலாம். எடு.கா: Imdb, amazon reviews தரவுக்கணங்கள் (datasets) amazon reviews dataset என்பது அமேசான் விற்பனைத் தளத்தில் வாங்கிய பொருள் பற்றி நுகர்வோர் எழுதிய பின்னூட்டங்களின் தொகுப்பு. எடுத்துக்காட்டுக்காக அத்தரவுக்கணத்தில் இருந்து ஒரு மாதிரியைப் படம்-1 இல் காண்க. இதுவொரு சொற்சுவை (sentiment detection) தரவுக்கணம்.

"I bought this for my husband who plays the piano. He is having a wonderful time playing these old hymns. The music is at times hard to read because we think the book was published for singing from more than playing from. Great purchase though!"	5/5
படம்-1: அமேசான் தரவுக்கணத்தில் இருந்து ஒரு மாதிரி . அமெசான் தளத்தில் வாங்கிய புத்தகத்தைப் பற்றி ஒருவர் எழுதிய புகழுரை பின்னூட்டம்.	

சொற்றொடர் புகழ்ச்சி நிரம்பி இருக்கிறதா , அல்லது இகழ்ச்சி நிரம்பி இருக்கிறதா என்று அடையாளமிடப்பட்ட தரவு. படம்-2 இல் பெயர்ச்சொல் உணரி தரவுக்கணத்தில் இருந்து ஒரு மாதிரி . சொற்றொடரில் வரும் பெயர்ச்சொற்களை அடையாளங்காட்டும் தரவு. இதே போல் சொல்வகை (Parts of Speech/POS) விவரத்தை அடையாளமிட்டு ஆங்கிலத்திற்குப் பல தரவுகணங்கள் உள்ளன . இப்படி ஒவ்வொரு வகையான அடையாளங்களைக் குறித்துக் காட்டி தரவுகணங்களை உருவாக்குவதன் நோக்கம், மொழியின் பல பரிமானங்களை கணினிகளுக்கு எடுத்துரைக்கவே . ஒரு கணக்கை இப்படித்தான் செய்ய வேண்டும் என்று அறுதியிட்டு நிரலாக கணினிக்குக் கட்டளையிடுவது போல நேரடியாக மொழியைக் கற்றுத்தர இயலாமையால் தான் இப்படி பல்வேறு அடையாளமிட்ட தரவுக்கணங்களை (annotated datasets) தொகுக்க வேண்டியுள்ளது. இப்படி சொற்சுவையறிதல் (sentiment detection), பெயர்ச்சொல்லுணர்தல் (named entity recognition), சொல்வகை குறித்தல் (POS tagging), கேள்விக்கு-பதிலுரைத்தல் (question answering) என பல்வேறு இயன்மொழியாள்கை வினையாட்டங்களுக்கு (natural language processing tasks) தரவுக்கணங்களை படைத்தல் அவசியம்.

கடைச்சங்க	காலமான	கி.மு.400 க்கும்	கி.பி.100 க்கும்	இடைப்பட்ட	காலத்தில்
-	-	காலம்	காலம்	-	-
திருவள்ளுவர்	திருக்குறளை	தமிழ்ச்சங்கத்தில்	அரங்கேற்ற	மிகவும்	சிரமப்பட்டதாகவும்
பெயர்	பெயர்	நிறுவனம்	-	-	-
முடிவில்	ஔவையாரின்	துணையோடு	மதுரையில்	அரங்கேற்றியதாகவும்	நம்பப்படுகிறது
-	பெயர்	-	ஊர்	-	-
படம்-2: "கடைச்சங்க காலமான கி.மு.400 க்கும் கி.பி.100 க்கும் இடைப்பட்ட காலத்தில் திருவள்ளுவர், திருக்குறளை தமிழ்ச்சங்கத்தில் அரங்கேற்ற மிகவும் சிரமப்பட்டதாகவும், முடிவில் ஔவையாரின் துணையோடு, மதுரையில் அரங்கேற்றியதாகவும் நம்பப்படுகிறது." என்ற சொற்றொடரின் பெயர்ச்சொல் சிட்டைகளை கீழே காண்க.					

1.3. **செய்யறிவும் கற்குங்கருவியியலும்**

பளுவைக் குறைக்க பல உத்திகளை மாந்தர் தம் வரலாற்றில் கையாண்டுள்ளனர். உயிரில்லா கருவிகளைப் படைத்தும், உயிருள்ள விலங்குகளை அரவணைத்தும் வேலைச்சுமையைக் குறைக்க முயன்றுள்ளனர். வேல்கம்பு, வில் அம்பு, சுத்தியல், மண்வெட்டி, ஏர்க்கலப்பை என பலகருவிகள் தொடங்கிப் பொதி சுமக்க , வெகுதூரப் பயணங்களுக்கு வண்டியிழுக்க , வீட்டைக் காவற்காக்க , வேட்டைக்கு உதவ என நாய் , கழுதை, மாடு, குதிரை முதலிய பல்வேறு விலங்குகளை வீட்டினமாக்கியது வரையில் பலவாறாக உடலுழைப்பைக் குறைக்க முயன்றுள்ளனர் . இதேபோல் மூளைச்சுமையை குறைக்க எழுத்து , ஏடு முதலிய கருவிகளையும் ; கடத்த, காக்க வேண்டிய தகவல்களைக் குறித்து வைக்க நெறி முறைகளையும், மொழிக் கூறுகளையும் படைத்துள்ளனர்.

இதில் கவனத்தில் கொள்ள வேண்டியவை இரண்டு . ஒன்று பொதுமையிலிருந்து விதுமை நோக்கி (generalization to specialization) பயணிக்கும் இயல்பு . காவலுக்காக வீட்டினமாக்கிய நாய், வேட்டைக்கென்று தனியாகவும், துப்பறிவதற்கு என்று தனியாகவும் இனப்பெருக்கம் செய்யப்பட்டது எளிமையான எடுத்துக்காட்டு. இரண்டாவது மூளைச் சுமையைக் குறைக்கும் எந்த உத்திகளும் உயிர் உள்ளவை கிடையாது . மாந்தரின் அறிவுத்திறனை எந்த விலங்கைக் கொண்டும் ஈடு செய்ய இயலவில்லை. பொதி சுமக்கும் கழுதை வேறெந்த வேலையையும் கற்காது . பயணத்திற்கு மாட்டுவண்டியில் பூட்டிய காளைகள் நிலத்தை உழலாம் , ஆனால் எத்தனை ஆண்டுகள் வேளாண் நிலத்தில் கிடந்தாலும் அது வேளாண்மை கற்காது.

இருபதாம் நூற்றாண்டு வரை இப்படித்தான் . அலான்சோ சர்ச்சு என்பாரும் , ஆலன் டூரிங் என்பாரும் கணினியியலை ஒரு துறையாக நிறுவிய பிறகே பொதுமையிலிருந்து-விதுமை விடுத்து பொதுமை நோக்கித் திரும்பியது அறிவியல் . கணினி என்ற ஒரே கருவியை ஒன்றுக்குமேற்பட்ட பல்வேறு பணிகளைச் செய்யுமாறு பணிக்கமுடியும் என்ற நிலையை எட்டியது கடந்த நூற்றாண்டில்தான். கணினியை தேவைக் கேற்ப பணிக்க அவற்றுக்கு என்ன செய்யவேண்டும் என்று அறிவுறுத்த வேண்டும். அப்படி அறிவுத்தும் முறைகளை இருபெரும் பிரிவுகளாக பிரிக்கலாம் . அ) நிரலாக்கம்(Programming) என்பது பணிகளின் செய்முறையை ஆய்ந்து என்னென்ன செய்யவேண்டும், எந்த வரிசையில் செய்யவேண்டும் என்று கட்டளைகளை எந்தவிதப் பொருள்மயக்கமும் (பொருள்மயக்கங்கள் தான் நிரல் வழுக்களுக்கு(programming bugs) முதற்காரணம்) இல்லாமல் தெளிவாக நிரலாய் தரித்துத் தருவது. ஆ) மாறாக ஒவ்வொரு பணியின் செய்முறைக்கும் நிரல் தரிக்காமல், மாந்தர் எப்படி கற்கின்றனரோ அப்படிக் கற்கச் செய்வது செய்யறிவாகும் (Artificial Intelligence). செய்யறிவில் கணினியோ /கருவியோ எப்படி கற்கிறது, என்ன கற்கிறது என்பதைப் பொறுத்துப் பல துறைகள் அதனுள் அடங்கும்.

அதில் ஒன்று தான் கற்குங்கருவியியல் (Machine Learning). இதையிதை இப்படித்தான் செய்யவேண்டும் என்று கட்டளைகளை எழுதாமல் , என்ன செய்யவேண்டும் என்பதை

எடுத்துக்காட்டுகளிலிருந்து பிழிந்தெடுக்கச் செய்வது கற்குங்கருவியியல் . இப்படிப் பிழிவதற்கான நுட்பங்களை புள்ளியியல், கணிதவியல் என்று பல்வேறு துறைகளிலிருந்து பெறலாம்.

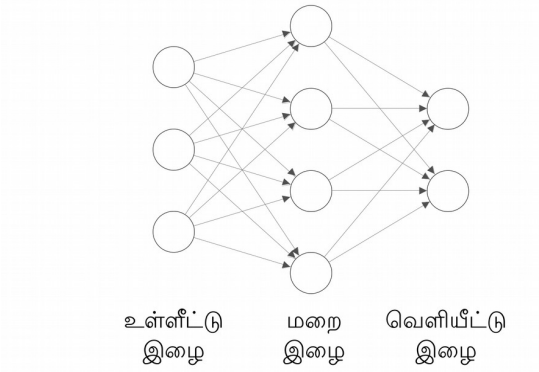
1.3.1. **நரவலை**

மூளையில் ஏறக்குறைய நூறுகோடி நரம்பணுக்கள் உள்ளன , ஒவ்வொரு அணுவும் மற்ற ஆயிரக்கணக்கான அணுக்களோடு பின்னபட்டிருக்கும் . இப்படியான பிணைப்புகள் கிட்டத்தட்ட நூறாயிரங்கோடி. கணினியில் எப்படி எளிமையான செயலாக்கம் கொண்ட டிரான்சிசுட்டர்கள் பல்லாயிர எண்ணிக்கையில் சேர்ந்து இயங்கும்போது கணினி வியத்தகு செயல்களைச் செய்கிறதோ அப்படியே எளிமையான நரம்பணுக்கள் கூட்டாக இயங்கும் போது மூளை வியத்தகு வேலைகளைச் செய்கிறது[1,2,3]. இயற்கையாக அமைந்த மூளையின் செயலையும் அமைப்பையும் உந்துதலாகக்கொண்டு செய்யறிவறிஞர்கள் படைத்தது தான் செய்நரவலை (Artificial Neural Network) சுருக்கமாக செ.ந.வ (ANN) நரவலை என்பது செய்யறிவு இயக்க கொள்கையில் ஒன்றான கூட்டியியக்கியத்தின் (connectionism) மெய்ப்படுத்தும் (realization) முறைகளில் ஒன்று . நரம்பணுக்கள்(neurons) ஒன்றோடொன்று பிணைந்து வலை போன்ற அமைப்பை ஆக்கும் . நரம்பணுக்கள் தனித்தனியே சிறிய சிறிய செயல்களைச் செய்தாலும் அவை கூட்டாக ஒருங்கிணைந்து இயங்கும் போது சிக்கல் மிகுந்த செயல்களைச் செய்ய வல்லன.

செயலிலும் சரி , அமைப்பிலும் சரி , செய்நரவலையிலுள்ள நரம்பணுக்கள் முற்றிலுமாக மூளையின் நரம்பணுக்களை ஒத்தவை அல்ல . அமைப்பு, செயல், பயிலும் விதம் முதலிய பல கூறுகளைப் பொறுத்து செய்நரவலைகளிலும் பல்வேறு வகைகளுண்டு . ஒவ்வொன்றிலும் நரம்பணுவின் பண்புகள் வெவ்வேறு அளவுகளில் மூளையின் நரம்பணுவை ஒத்ததாக இருக்கும் . இங்கு நாம் காணப்போகும் நரம்பணுவோ மிகமிக எளிமையான ஒன்று.

நாம் காணப்போகும் நரவலைகள் இழையிழையாக அமைந்திருக்கும் . ஒரு இழையில் பல அணுக்கள் இருக்கும். ஒவ்வொரு இயிலுள்ள அணுக்கள் அடுத்துள்ள இழையிலிருக்கும் அனைத்து அணுக்களோடும் பிணைந்திருக்கும் . எல்லாப் பிணைப்புகளும் ஒரே வலிமையுடைன அல்ல . பிணைப்புகளின் வலிமைகளை மாற்றுவதால் , ஒரே அமைப்புடைய ஒரி வேறு நரவலைகள் வெவ்வேறு பணிகளைச் செய்ய வைக்கமுடியும்[4,5,6,7,8,9]. முதல் இழை உள்ளீட்டு இழை (input layer) என்றும், கடையிழை வெளியீட்டு இழை (output layer)என்றும், இடையில் இருக்கும் இழைகள் மறையிழைகள் (hidden layers) என்று அழைக்கப்படும் . எல்லா இழைகளும் ஒரெயளவிலான எண்ணிக்கையில் அணுக்களைப் பெற்றிருக்க வேண்டும் என்பதில்லை . காட்டாக மூன்று இழைகள் இருக்கும் வலையைக் கீழ்வரும் படம் -3 இல் காணலாம் . முதல் இழையில் மூன்று அணுக்களும் , இரண்டாம் இழையில் நான்கு அணுக்களும் , கடை இழையில் இரண்டு அணுக்களும் இருப்பதை நோக்குக. உள்ளீட்டுத் தரவு உள்ளீட்டிழையில் தொடங்கி பல இழைகளைக் கடந்து வெளியீட்டிழையில்

வந்து சேரும் போது சரியான வெளியீடாக இருக்க வேண்டும் . அப்படி இல்லையெனில் சரியான வெளியீட்டிற்கும், வலை கணித்த வெளியீட்டிற்கும் உள்ள இடைவெளியைப் பிழையென (error) கருதுவோம். பிழையை பின்னோக்கிப் பாய்ச்சி இழைகளுக்கு இடையில் உள்ள இணைப்புகளை வலுப்படுத்தவோ, நலிவுறவோ செய்வோம். இப்படி பிழையைப்பாய்ச்சி வலையைத் திருத்துவதால் , (பின்)பிழைபாய்ச்சல் நரவலை (backpropagation neural networks) [10,11] என்று அழைக்கப்படும். இப்படி பிழைபாய்ச்சி வலையிலுள்ள பிணைப்புகளை மாற்றுவதை திருத்துதல் அல்லது பயிற்றுவித்தல்(training). இங்கிருந்து நரவலை என்று சொல்வதெல்லாம் பிழைபாய்ச்ச நரவலையையே குறிக்கும்.



படம்-3: மூவிழை நரவலை

நரவலை உள்ளீடாக எண்களின் பட்டியலையே கொள்ளும் . வெளியீடாகவும் எண்களின் பட்டியலே வரும். இப்படி எண்களின் பட்டியலை கணிதத்தில் காவி என்று சொல்வர். காவியைப் பற்றி விவரம் பின்னே வருகிறது. இங்கே கவனிக்க வேண்டியது மூன்று அணுக்கள் உள்ள இந்த வலை மூன்று எண்களுள்ள காவியை உட்கொள்ளும். மூன்று எண்கள் என்றால், முப்பரிமான வெளியில் (three dimensional space) உள்ள ஏதோ ஒரு புள்ளியைக் குறிக்கும் . அதே போல வெளிவரும் இருபரிமானக் காவிகள் இருபரிமான வெளியில் உள்ள புள்ளிகளைக் குறிக்கும் . இடையிலுள்ள இழைகளுக்கும் இதை பொறுத்திப் பார்த்தால் , நரவலை ஒரு வெளியில் உள்ள புள்ளிகளை வெவ்வேறு வெளிகளுக்குக் கடத்துகிறது. பிழையைப் பாய்ச்சுவதின் மூலம் இந்த கடத்தல் நடக்கும் பாதைகளை கட்டமைப்பதை உணரலாம்.

அமைப்பைப் பொறுத்து நரவலைகளை இரண்டாக பகுக்கலாம் . முன்னூட்ட நரவலை (feedforward neural network), பின்னூட்ட நரவலை (recurrent neural network) [12,13]. முன்னூட்ட வலையில் மடைகள் (loop) ஏதுமில்லாமையால் தகவல்/உள்ளீடு ஒரே திசையில் இழைகளின் ஊடாக ஒருமுறை தான் பயணிக்கும். பின்னூட்ட வலையிலோ மடையின் துணையால் உள்ளீடு இழைகளின் ஊடாக பலமுறை பயணிக்கக் கூடும். இயன்மொழி வினையாட்டங்களில் பின்னூட்ட நரவலைகளின் பங்கு

முக்கியத்துவம் வாய்ந்தது. ஒரே வலை, ஒரு தொடரில் உள்ள சொற்களை ஒவ்வொன்றாக உள்வாங்கி தொடரின் பொருளை அறியும் . பின்னூட்ட நரவலையின் மடைப்பண்பு முன்வந்த சொற்களை நினைவில் கொண்டு எதிர் வரும் சொற்களை நோக்க வழிசெய்கிறது.

2. ஒப்புகளும் முறைகளும்

இயன்மொழியாள்கை பிரிவில் சொன்னது போலப் பல வினையாட்டங்களை பற்றிப் பார்த்தோம். அடையாளங்களிட்ட தரவுக்கணங்களை உருவாக்குவது எளிதான காரியமல்ல . ஆகையால் முதல் படியாக அடையாளம் ஏதுமில்லாமல் செய்யக்கூடிய வினையாட்டம் ஒன்றைச் செய்கிறோம். மொழியொப்பாக்கம்.

2.1 மொழியொப்பு

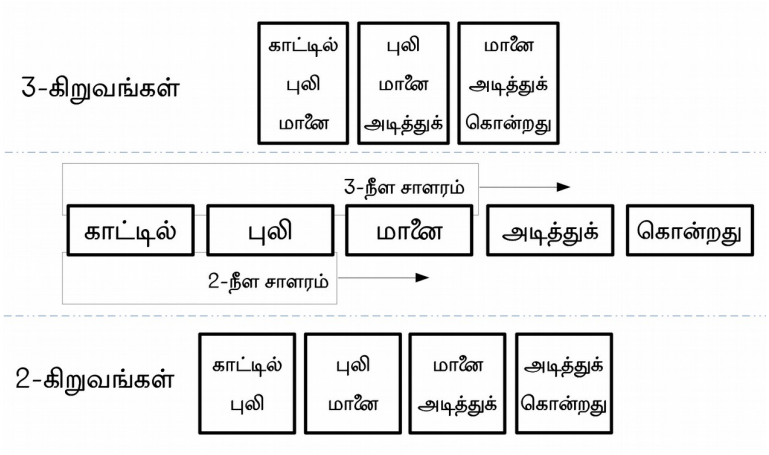
ஒரு கேள்வி. காட்டில் புலி ____ அடித்துத் தின்றது. விடுபட்ட சொல் என்ன? நம் மனதில் முதலில் தோன்றும் சொல் மானை. சிலருக்கு ஆட்டை, குதிரையை என்ற சொற்களும் தோன்றலாம். மான் முதலிய விலங்குகளை புலி முதலிய விலங்குகள் அடித்துக் கொள்ளும் என்ற உலகைப்பற்றிய அறிவு இருப்பினும் மான் என்று தோன்றாமல் மானை என்று தோன்றக்காரணம் , தமிழில் இப்படித்தான் சொற்றொடர் அமையும் என்று நமக்குள்ள மொழியாளுமை.

மாந்த மூளை எப்படி மொழியைப் புரிந்துகொள்கிறது என்பதை இன்னும் ஐயத்திற்கு இடங்கொடாமல் அறுதியிட்டுச் சொல்வதற்கில்லை . இன்னும் ஆய்வில் உள்ளப் புலம் தான் அது . எனினும் இதிலிருந்து ஒன்றை உய்த்தறியலாம் . சுற்றி அமைந்தச் சொற்கள் (புலி, அடித்து, தின்றது) விடுபட்டச் சொல் (மானை) என்னவாக இருக்கலாம் என்று கண்டறிய உதவுகிறது . இதே சோதனை வேறுமாதிரி செய்வோமானால், புலி மானை அடித்து _____. இப்போது விடுபட்டச் சொல் என்ன என்று கேட்டால் கொன்றது, தின்றது என்ற சொற்கள் தோன்றலாம். இப்படி ஒரு சொற்றொடரில் இருக்கும் சொற்களை கொண்டு விடுபட்ட சொற்களைக் கணிக்கச்செய்வது மொழியொப்பாக்கம் எனப்படும் . புள்ளியியல், கற்குங்கருவியியல் நுட்பங்களைக் கொண்டு மொழியின் கூறுகளை கணியொப்பாக்குவதே கணிய-மொழி-யொப்பாக்கம் (computational language model).

2.1.1. எ-கிறுவம்

மேற்சொன்ன படி முன்வரும் சொற்களைக் கொண்டு தொடர்ந்து வரும் சொல்லைக் கணித்தலின் பெயர் , எ-கிறுவ மொழியொப்பாக்கம். முன்வரும் சொற்களில் அனைத்துச் சொற்களையும் கருத்தில் கொண்டு அடுத்த சொல்லைக் கணிக்கவேண்டும் என்றில்லை . (புலி, மானை, அடித்து) என மூன்று சொற்களைக் கொண்டு கணித்தால் 4-கிறுவம் எனக்கொள்க. (மானை அடித்து) என இரண்டு சொற்களை மட்டும் கருத்தில் கொண்டு கணித்தால் 3-கிறுவம் எனக்கொள்க.

இப்படி எத்தனைச் சொற்களைக் கணக்கில் கொள்கிறோமோ அதைப் பொறுத்து எ-கிறுவம் எனலாம்.



படம்-4: கிறுவங்களை எப்படி ஒரு சொற்றொடரில் இருந்து படைக்கலாம் என்பதின் ஒரு உத்தி. பல்வேறு நீளங்கொண்ட சாளரங்களி சொற்றொடரின் மீது ஓட்டி, சாளரத்தினுள் அடங்கும் சொற்கள் கிறுவங்களாகும்.

2.1.2. **தாவிய-கிறுவம் 4**

எ-கிறுவத்தில் முன்வரும் சொற்களை அதாவது ஒரு சொற்றொடரின் முன்பகுதியைக் கொண்டு மீதியைக் கணித்தோம். எடு.கா. புலி மானை அடித்து தின்றது என்ற சொற்றொடரில் உள்ள அனைத்து இருகிறுவங்களையும்(2-கிறுவங்கள்) தொகுத்தால் [புலி மானை, மானை அடித்து, அடித்து தின்றது] என்பன. [புலி அடித்து] என்பது அச்சொற்றோடருக்கான இருகிறுவமில்லை. ஏனெனில் தொடர்ச்சி அறுந்து போகிறது . தாவிய-கிறுவத்திலோ முன்வரும் சொற்கள் தொடர்சியாக் இருக்கவேண்டும் என்றில்லை. தாவிய-கிறுவம் *(skip-gram)* என்பது எ-கிறுவத்தின் (பொதுமைப் படுத்திய, generalization) நீட்சி. த-தாவிய-எ-கிறுவம் என்றால் எ -எண்ணிக்கையுள்ள கிறுவங்களில் அதன் உறுப்புகள் சொற்றொடரில் த-தூரத்தில் இருக்கும் எனக்கொள்க. எனவே [புலி அடித்து] என்பது சொற்றொடரின் *1*-தாவிய-*2*-கிறுவங்களில் ஒன்று.

கிறுவங்களை திரட்ட ஒரு சொற்றொடரின் மீது நகரும் சாளரத்தை ஒட்டி சாளரத்தினும் வரும் உருப்படிகளை (சொற்களை) கிறுவங்களாக கொள்ளலாம். சாளரத்தின் நீளம் இரண்டாய் இருப்பின் இருகிறவம் கிடைக்கும் , மூன்றாய் இருப்பின் மூகிறுவம் கிடைக்கும் . இப்படி த -தாவிய-எ-கிறுவங்களை, சேகரித்த பனுவல்களில் இருந்து திரட்டி , நரவலைக்கு எடுத்துக்காட்டுகளாக ஊட்டித் தான் கணியொப்பை வடிக்கிறோம்.

2.2. **கணியொப்புகள்**

பெரிய கட்டங்களைக் கட்ட தொடங்கும் முன்பு தாள்கள் அட்டைகள் கொண்டு கட்டட அமைப்பைப் சிறிய வடிவில் அச்சு /போலி பாவணைச் செய்வது கட்டடப் பொறியாளரின் வழக்கம் .

இப்படி அளவிலும் வடிவிலும் மட்டும் சிறியதான பாவணைச் செய்ய வேண்டும் என்றில்லை , செயலிலும் இருக்கலாம். இயற்கையின் பல நிகழ்வுகள் சிக்கல் மிகுந்தவை , எளிமையாக புரிந்து கொள்ளத்தக்கன அல்ல. இப்படி இயற்கையின் நிகழ்வுகளின் இயல்புகளைத் துல்லியமாக ஆய்ந்தறிய முடியாத இடங்களில் தோராயாமாகவாது அறிய உதவும் முறை ஒப்பாக்கம் (modelling). ஒப்பாக்கம் கணினியின் துணைகொண்டு வடிக்கப்பட்டால் அது கணியொப்பு(computational model).

வரிசை எண்	சொல்	நிகழ் வெண்		வரிசை எண்	சொல்	நிகழ் வெண்
1	இப்படியான	1		1	முழுக்க	2
2	ஏறக்குறை	1		2	இப்படியான	1
3	ஒப்புகளை	1		3	ஏறக்குறை	1
4	கற்குங்கருவியியல்	1		4	ஒப்புகளை	1
5	நுணுக்கங்களை	1		5	கற்குங்கருவியியல்	1
6	பற்றியதே.	1		6	நுணுக்கங்களை	1
7	முழுக்க	2		7	பற்றியதே.	1
8	வடிப்பதற்கான	1		8	வடிப்பதற்கான	1
	(அ)				(ஆ)	

அட்டவணை-1: சொல்லடைவு எடுத்துக்காட்டு.

கற்குங்கருவியியல் துறை ஏறக்குறைய முழுக்க முழுக்க இப்படியான ஒப்புகளை வடிப்பதற்கான நுட்பங்களைப் பற்றியதே. அதில் மொழியாய்வுக்கு முக்கியமான ஒன்று சொற்றூவல் (சொல்-தூவல், word-embedding). கணியொப்புகள் உட்கொள்ளும் வகையில் சொற்களை எண்களாக மாற்றுவது மிக அவசியம்.

அ) எளிமையான அதேவேளை சாரம் ஏதுமற்ற முறையில் சொற்களை எண்ணாக்கும் வழி , இருக்கும் சொற்களனைத்தையும் ஒரு பட்டியலாக (அகர)வரிசைப் படுத்தி சொல்வரும் வரிசையை அச்சொல்லின் எண் வடிவாகக் கொள்ளலாம் . இப்பத்தியின் முதல் வரியில் இருக்கும் சொற்களை மட்டும் தொகுத்தால் கிடைக்கும் சொல்லடைவை*(vocabulary)* அட்டவணை-1-இல் காணலாம்.. முழுக்க என்ற சொல் மட்டும் இரண்டு முறை வந்திருப்பது கவனிக்கத் தக்கது . சொற்களை அகரவரிசைப் படித்தான் வரிசைப்படுத்த வேண்டும் என்றல்ல, அவை எத்தனை முறை வருகின்றன /நிகழ்கின்றன என்பதைப் பொறுத்தும் இருக்கலாம்.

ஆ) இன்னொரு முறை காவிகளாக மாற்றுவது. காவி*(vector)* என்பது ஓர் எண்ணாக இல்லாமல் எண்களின் பட்டியலாக மாற்றுவது. இங்கே பட்டியலின் நீளத்தைக் காவியின் பரிமானம் எனக்கொள்க. காவி பல பரிமானங்களுடையது. எடு.கா: *[0 0]* இருபரிமான காவி, *[3 2 4]* என்பது முப்பரிமான காவி,

இப்படி நூறு அல்லது ஆயிரம் எனப் பல அளவுகளிலான காவிகளை பயன்படுத்துவது கற்குங்கருவியியலில் வாடிக்கையான ஒன்று . *[0 1], [1 0]* என்ற காவிகள் இரண்டும் இருபரிமானமுடையது என்றாலும், இரண்டிலும் 0-உம், 1-உம் தான் வருகிறது என்றாலும் , இரண்டும் வெவ்வேறு காவிகள் என்பதைக் கருத்தில் கொள்க . எண்களின் மதிப்பும் , அவை இருக்கும் இடங்களும் முக்கியத்துவம் வாய்ந்தவை என்பதையுணர்க.

எளிமையான வரிசை எண்ணை விடுத்து எதற்கு இப்படிப் பலபரிமானக் காவியைப் பயன்படுத்த வேண்டும் என்ற கேள்வி எழுவது இயல்பு . காவிகள் பல பரிமானங்களில் வந்தாலும் , சொற்களைக் காவிகளாக மாற்றும்போது எல்லாச்சொற்களும் ஒரே அளவிலான காவிகளைக் கொண்டுதான் குறிக்கப்படும் . வெவ்வேறு சொற்களைக்குறிக்க வெவ்வேறு காவிகள் இருக்கும் , அதாவது ஒவ்வொரு சொல்லும் ஒரே அளவிலான பரிமானங்கொண்ட காவிகளைக் கொண்டு குறிக்கப் பெற்றாலும், காவிக்குள் உள்ள எண்களின் மதிப்பு சொற்களை வேறுபடுத்திக்காட்டும். இப்படி காவிகளின் அளவு நிலையானதாக இருப்பதின் சாதகம் , மொழியின் அதிலும் குறிப்பாக தமிழ் போன்ற ஒட்டுநிலை மொழியின் சொல்லடைவை வரிசையெண் கொண்டு குறியிடும் போது வரும் சிக்கலுடன் ஒப்பிடும் போது வெட்ட வெளிச்சமாகும் . ஆனால் காவிகளை அகரவரிசை கொண்டோ , நிகழ்வெண் வரிசை கொண்டோ நாமாக ஒரு காவியை ஒரு சொல்லுக்கு கொடுக்க முடியாது . சொல்லுக்காண காவிகளைக் கண்டறிய வேண்டும் . காவிகளை நரவலையின் ஓர் உறுப்பாய் அமைத்துவிட்டால், திரட்டிய கிறுவங்களைப் பொறுத்து வலையை திருத்துவதின் மூலம் காவிகளைக் கண்டறியச் செய்யலாம். இப்படிச் செய்வதின் முலம் ஒவ்வொரு சொல்லுக்கும் ஒரு காவி , அதாவது காவியின் பரிமான வெளியில் ஒரு புள்ளி ஒதுக்கப்படுகிறது . அதாவது 200-பரிமான காவிகளை சொற்களுக்கு ஒதுக்குகிறோம் என்றால் , 200-பரிமான வெளியில், சொற்களை தூவி விடுகிறோம் எனக்கொள்ளலாம். ஆகையால் சொற்களைக் குறிக்கும் காவிகள் , சொற்றூவல் (சொல்தூவல்) காவிகள் என்றும் அழைக்கப்படும்.

3. **தரவுக்கணம்**

3.1. **தரவுச்சேகாரம்**

இருவேறு வழிகளில் சொல்லோடை -2021 தரவுகள் (பனுவல்கள்) சேகரிக்கப்பட்டன. அ) ஏற்கனவே தொக்குக்கப்பட்ட தரவுகணங்கள். AI4Bharat, Leipzig Tamil Corpus, tamiltext-7M.txt

ஆ) இணையத்தில் வெளிவரும் பல்வேறு செய்தித்தாள்களிலும், வலைப் பூக்களிலும், விக்கிப்பீடியா போன்ற களஞ்சியங்களுலும் இருந்து , தமிழ் பனுவல்களைச் சேகரித்தோம் . இணையதளத்தில் இருந்து பனுவல் படங்கள் முதலிய வளங்களை கவர எழுதப்படும் நிரல்வகை சிலந்தி எனப்படும் . இணையதளப் பக்கங்கள் ஒன்றோடொன்று தொடுக்கப்பட்டிருப்பதால் , ஒவ்வொரு பக்கத்தையும்

அணுகி அவற்றில் இருக்கும் வளத்தைக் கவர்வதால் சிலந்தி என்ற பெயர் சாலப் பொருந்தும் . ஒவ்வொரு தளமும் வெவ்வேறு வடிவங்களைக் கொண்டுள்ளன . தளத்திற்கேற்பச் சிலந்திகளை எழுத Scrapy என்ற பைத்தான்ச் (python) சட்டகத்தைக் (framework) கையாள்கிறோம். இப்படி கவர்ந்த பக்கங்களை mongodb எனும் தரவுதளத்தில் சேமிக்கிறோம். கவரப்பட்டப் பக்கங்களில் இருந்து கட்டுரை, அதன் தலைப்பு , வெளிவந்த நாள், தொடர்பான சிட்டைகள், உரையின் ஆசிரியர் முதலிய மேந்தரவுகளும் (மேல் விவரங்கள் , அதாவது தரவைப்பற்றிய தரவுகள்) அடங்கும். எனினும் சொல்லோடை-2021 இல் முன்சொன்ன மேந்தரவு (metadata) அடங்காது. எதிர்காலத்தில் சரியாக ஒழுங்குபடுத்தி வெளியிடுவோம் . கட்டுரைகளை எப்படி உடைத்துப் பிரித்துச் சொற்றோடர்களாக மாற்றுகிறோம் என்பதைப்பற்றி கீழே காண்க.

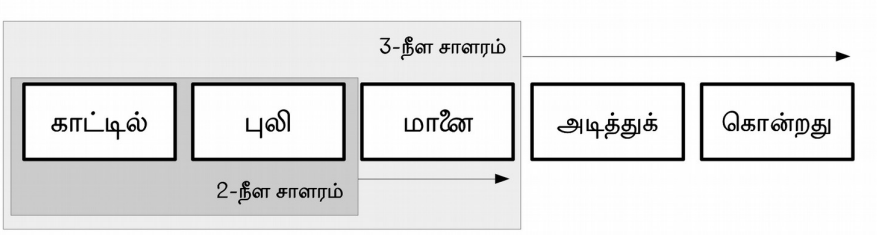
3.2. **செம்மைபடுத்துதல்**

அ) தொடரறுத்தல்: பைத்தான் மொழியின் NLTK களஞ்சியத்தின் sent_tokenize நிரல் கூற்றை கொண்டு சொற்றொடர்களை உடைக்கிறோம். ஆங்கில மொழிக்காக எழுதபட்டப் நிரல்கூறு எனினும், சிலவிடங்கள் தவிர்த்து (காட்டாக சொற்சுருக்கங்கள்:- த.நா. ; ஐ.நா) தமிழ் சொற்றொடர்களை உடைக்கவும் உதவுகிறது. இத்தரவுக் கணத்தில் இருப்பது போல் மாபேரளவில் பனுவல்கள் இருக்கும் போது, sent_tokenize கொண்டு தொடரறுத்தாலும்„ போதுமான அளவிற்கு நல்ல, முழுமையான தமிழ் சொற்றொடர்கள் கிடைக்கும் என்பது எங்கள் துணிபு . ஆ) வேற்றுமொழித் தொடர் நீக்கல் *:* தமிழும் ஆங்கிலமும் அல்லாத வேற்றுமொழி சொற்கள் , எழுத்துக்கள் நிரம்பிய சொற்றொடர்களை நீக்கிவிடுகிறோம். இ*)* முழுமையான ஆங்கிலத் தொடர்களை நீக்கல் *:* ஒரு சொற்றொடரில் முழுவதுமாக ஆங்கிலச்சொற்கள் மட்டுமே நிரம்பியிருப்பின் , அத்தொடர்களை நீக்கிவிடுகிறோம். ஈ) ஆங்கிலச் சொற்களை மறைத்தல*:* தமிழ் சொற்றொடர் களஞ்சியம் என்பதால் ஆங்கிலச்சொற்கள் வரும் இடத்தை மட்டும் சுட்டும் வகையில் , ஆங்கிலச்சொற்கள் வரும் இடங்களிலெல்லாம் , *#*ஆங்கிலம*#* என்ற சிறப்பு சொல்துண்டை*(special token)* அச்சொற்களுக்குப் பதிலாக வைக்கிறோம். உ) எண்களை மறைத்தல்: ஆங்கிலச்சொற்களை அடையாளம் காட்டியது போலவே , எண்களையும் *#*எண்*#* என்ற சிறப்பு சொல்துண்டாக மாற்றுகிறோம். ஊ) வரிசைப் படுத்தி நகல் நீக்கல் : இறுதியாக அனைத்துச் சொற்றொடர்களையும் வரிசைப்படுத்தி, நகல்களை நீக்க நமக்குக் கிடைக்கும் தொகுப்பு சொல்லோடை-*2021* தரவுக்கணம்.

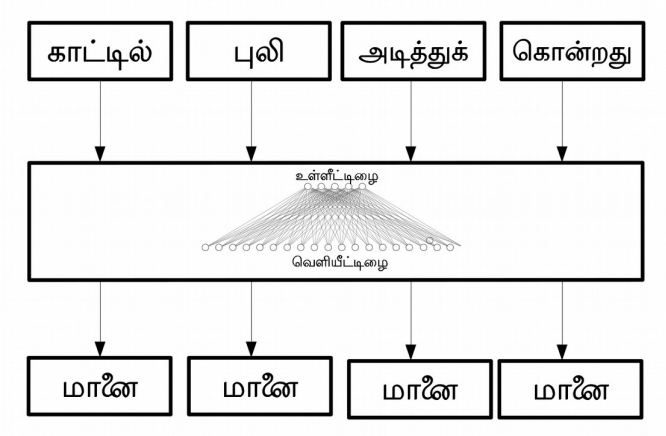
4. சோதனைகளும் முடிபுகளும்

சொற்றூவல்களை கொண்டு சொற்களின் சில இலக்கணப் பண்புகளைக் கண்டறியலாம் . ஈரிழை கொண்ட முன்னூட்ட நரவலையைக் கொண்டு (two layered feedforward neural network) மொழியொப்பை வடிக்கிறோம். தாவிய-கிறுவங்களை கொண்டு வடிக்கப்படுவதால் இதை தாவிய -

கிறுவ மொழியொப்பு என்றும் அழைக்கலாம். சொற்றொடர்களில் இருந்து கிறுவங்களை எடுக்க நகரும் சாளரம் என்ற உத்தியைக் கையாள்கிறோம். ஒரு நேரத்தில் சொற்றொடரில் இத்தனை சொற்களைத்தான் கருத்தில் கொள்ளவேண்டும் என்போமானல், அதை ஒரு குறிப்பிட்ட அளவிலான சாளரத்தின் ஊடாக பார்ப்பதாய் உருவகம் செய்துகொள்ளலாம். அச்சாளரத்தைச் சொற்றொடரின் மீது ஓட்டி கண்ணில் படும் சொற்களை எடுத்துக்கொண்டால், கிடைப்பது கிறுவம். சாளரத்தின் நீளத்தைப் பொறுத்து கிறுவத்தின் நீளமும் அமையும். 2-நீளச் சாளரம் இருகிறுவங்களைத் தரும், 10-நீளச் சாளரம் 10-கிறுவங்களைத் தரும். இந்த உத்தியையே கொஞ்சம் மாற்றி, இரண்டு நகரும் சாளரங்களை ஒன்றனுள் ஒன்றாக ஓட்டினால் தாவிய-கிறுவம் எடுக்கலாம். த-நீளமுள்ள சாளரத்தினும் எ-நீளமுள்ள சாளரத்தை ஓட்டினால், (த-எ)-தாவிய-எ-கிறுவம் கிடைக்கும். எடுத்துக்காட்டாக 3-நீளங்கொண்ட சாளரம் தாவலைக் கட்டுப்படுத்தும். அதனுள் 2-நீளங்கொண்ட சாளரத்தை ஓட்டினால் 1-தாவிய-2-கிறுவங்கள் கிடைக்கும்.



படம்-5: இரு நகரும் சாளரங்களைக் கொண்டு ஒன்றை ஒன்றனுள் ஓட்ட தாவிய-கிறுவங்கள் கிடைக்கும். மேற்காணும் இரண்டு சாரளங்களில் 3-நீளங்கொண்ட சாளரம் தாவலை கட்டுப்படுத்தும். அதனும் 2-நீளங்கொண்ட சாளரத்தை ஓட்டினால் 1-தாவிய-2-கிறுவங்கள் கிடைக்கும்.



படம்-6: ஈரிழை கொண்ட நரவலைக்கு சொற்றூவல்களை உள்ளீடாகக்கொடுத்து அருகமை கிறுவச் சொற்களை கணிக்க பணித்தால், பயிற்ச்சிக்கு பிறகு சொற்றூவல்கள் சொற்களின் சில இலக்கணப் பண்புகளை அகப்படுத்தி இருக்கும்.

வாடிவாசல்	ஜல்லிக்கட்டு	காளைகள்	சிரிக்கிறார்	மெரினா	காவிரி	பாயும்
கிரிக்கெட்	தோனி	டெஸ்ட்	போட்டி	அணி	பாகிஸ்தான்	பிசிசிஐ
கவிஞர்	கமல்	துரைசாமி	கருணாநிதி	ராமக்கிருஷ்ணன்	யுவன்	வழங்கியவர்
அழகான	அழகிய	அழகாக	பிரம்மாண்டமான	அழகு	டிரெண்ட்	காதல்

அட்டவணை-2: இரு சொற்களுக்கு இடையே இருக்கும் cosine-தூரத்தை வைத்து நெருங்கிய சொற்கள் என்னென்ன என்ற அலசலில் கிடைத்த எடுத்துக்காட்டுகள்.

படம்-6 நரவலையின் அமைப்பை விளக்கும் . சொற்றூவல் காவிகளின் அளவு *=200.* அதாவது உள்ளீட்டிழையின் பரிமானம் 200, மொத்தச் சொற்களின் எண்ணிக்கை (= 301515) வெளியீட்டிழையின் பரிமானமாகும். நரவலையை பயிற்றுவிக்கப் பட்ட பிறகு அதன் சொற்றூவல் காவிகளை எடுத்து அவற்றுள் இருக்கும் பண்புகளை ஆய்ந்தால் சுவையான சில விவரங்கள் புலப்படும் . Cosine-distance என்ற இரு காவிகளுக்குள் இருக்கும் தூரத்தைக் கொண்டு அட்டவணை -2 இல் உள்ள எடுத்துக்காட்டுகள் நிரப்பப் பட்டுள்ளன . வாடிவாசல் என்ற சொல்லின் காவியை மற்றனத்து சொற்களின் காவிகளோடு cosine-தூரத்தைக் கொண்டு ஒப்பிட்டு அருகில் இருக்கும் சொற்கள் என்னென்ன என்று பார்த்தால் ஜல்லிக்கட்டு , காளைகள், மெரினா முதலிய சொற்கள் நமக்குக் கிடைக்கின்றன. இது எப்படியான இலக்கணக் கூறுகளைச் சுட்டுகிறது என்று நேரடியாக புரியாவிட்டாலும், பார்க்கையில் தொடர்புடைய சொற்களை அறியத்தருகிறது.

4.1. **சொல்லோடை-2021 தரவுக்கண விவரம்**

சொற்றொடர் அடைவு : மேற்சொன்ன படி பனுவல்களை தொடர்களாக அறுத்து , பிழை களைந்து, வரிசைப்படுத்தி நகல்நீக்கி படைக்கப்பட்ட சொற்றொடர் தரவுக்கணம், சொல்லோடை-2021. மொத்தம் ஏழு கோடிக்கும் சற்றே அதிக (72000000) சொற்றொடர்களைக் கொண்டுள்ளது.

சொல்லடைவு: எல்லாச் சொற்றொடர்களையும் தொடருடைத்து (word-tokenize), குறைந்தது நூறு முறையாவது உரைகளில் வந்திருக்கும் சொற்களை மட்டும் கொண்டு சொல்லடைவு ஒன்றை படைக்கிறோம். இதில் மொத்தம் முன்னூறாயிரம் சொற்களுக்கும் கொஞ்சம் கூடுதலாக உள்ளன . சொல்லடைவு அகரவரிசையிலும், நிகழ்வெண் வரிசையிலும் வழங்குகிறோம்.

கிறுவடைவு: கிறுவங்கள் சொல்லளவில் தான் இருக்க வேண்டும் என்றில்லை , எழுத்தளவிலும் இருக்கலாம். சொல்லடைவை முதலாகக்கொண்டு எழுத்தளவிலான கிறுவங்கள் ஆக்குகிறோம் . தமிழின் உயிர் மெய் புணர்ச்சி இயல்பால் இருவகையான எழுத்துக் கிறுவங்களை படைக்கலாம் . அ) உயிர்மெய் கிறுவம்: ஒரு சொல்லை அப்படியே எடுத்து கிறுவம்பிரித்தால் 1-கிறுவம் [சொ ல் லை] எனவும், 2-கிறுவம் [சொல் ல்லை] எனவும், 3-கிறுவம் [சொல்லை] எனவும் வரும் . 4-கிறுவம்

இச்சொல்லுக்கு இல்லை என்பது தெளிவு. அ) மெய்யுயிர் கிறுவம்: சொல்லை மெய்யும்-உயிருமாக *[ச் ஒ ல் ல் ஐ]* தனித்தனியாக கிறுவம்பிரித்தால் 1-கிறுவம் *[ச் ஒ ல் ல் ஐ]* எனவும் 2-கிறுவம் *[சொ, ஒல், ல்ல், லை]* எனவும் வரும். இப்படிப் பிரிப்பதின் நோக்கம் புணர்ச்சியின் காரணமாக மறைந்திருக்கும் சொற்களை கடைந்தெடுப்பதுவே. எனினும் மெய்யுமுயிருமாக பிரிக்கும் போது முதல் எழுத்தையும் , கடையெழுத்தையும் பிரிக்காமல் அப்படியே வைத்துவிடுகிறோம் . இது ஒற்றெழுத்தில் தொடங்கும் , உயிரில் முடியும் கிறுவங்களை குறைத்துவிடுகிறது . ஒன்று முதல் பத்தாம் கிறுவம் வரை உயிர்மெய் வகையிலும், ஒன்று முதல் பதினாறாம் கிறுவம் வரை மெய்யுயிர் வகையிலும் கிறுவம்பிரிக்கிறோம் . மெய்யுயிர் கிறுவத்தில் வரும் மூங்கிறுவங்கள் உயிர்மெய் கிறுவத்தில் வரும் இருகிறுவங்களில் வரும், என்பது குறிப்பிடத் தக்கது . ஆகையினால் மெய்யுயிர் கிறுவங்களை முழுதாக (1...16) படைத்த பின்பு, அனைத்தையும் ஒன்றாக்கி மீண்டும் கிறுவத்தின் நீளம் பொறுத்து 1-கிறுவம், 2-கிறுவம் எனப் பிரித்து தனித்தனி கோப்புகளில் சேமிக்கிறோம். இப்படி கிறுவம்பிரித்ததற்கு எந்த நோக்கமுமில்லை. இது எந்த வகையில் பயன்படும் என்று சொல்வதற்கில்லை . எனினும் அனைத்து ஆய்வாளர்களுக்கும் கிட்டும் வகையில் வெளியிட்டால் புதிய சிந்தனைகளும் பயன்பாடுகளும் தோன்றும் என்பது எங்கள் எதிர்பார்ப்பும் நம்பிக்கையும்.

5. **முடிவுரை**

இப்பணியில், செய்யறிவுத்துறை என்பது கணினியியல், நரம்பியல், மொழியியல், உளவியல், முதலிய பலப்பல துறைகளோடு பின்னிப்பிணைந்தது. முடிந்தவரை எளிமையான முறையில் ஏன் தமிழை செய்யவறிவுத் துறை அடியெடுத்து வைக்கவேண்டும் என்பதன் அவசியத்தை உணர்த்த முயன்றுள்ளோம். செய்யறிவிற்கும் கற்குங்கருவியியலுக்கும் தரவுக்கணங்கள் எப்படி இன்றியமையாதவை என்பதை அலசுகிறோம் .. எவ்வித அடையாளக்குறிகளும், சிட்டைகளும் இல்லாத வெறும் பனுவல்களைக் கொண்ட தரவுக்கணங்களை வைத்தே சிறப்பான செயலாக்கங்களை செய்யமுடியும் என்பதையும் தாவிய-கிறுவ ஒப்பின் மூலம் சுட்டிக் காட்டியுள்ளோம். வருங்காலத்தில் பல்வேறு அடையாளச்சிட்டை போன்ற மேந்தரவு ஏறிய தரவுக்கணங்களை படைப்பது மிகத்தேவையான் ஒன்றாகும் . தமிழுக்கென இருக்கும் சிறப்பியல்புகளை கணினிக்குக் கற்றுத்தர மொழியற்ஞர்கள் பல்வேறு வினையாட்டங்களை வடிவமைக்க வேண்டும் என்று கோரிக்கை வைக்கிறோம் . தொகுத்த தரவுக்கணத்தையும், ஆக்கிய தாவிய-கிறுவ ஒப்பையும் பொதுவெளிக்கு சமர்ப்பிக்கிறோம்.

References

1. McCulloch, W. S.; Pitts, W. A Logical Calculus of the Ideas Immanent in Nervous Activity. The Bulletin of Mathematical Biophysics 1943, 5 (4), 115–133. https://doi.org/10.1007/bf02478259.

2. Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65(6), 386–408. https://doi.org/10.1037/h0042519
3. Parker, D. B. Learning-Logic : Casting the Cortex of the Human Brain in Silicon.; Massachusetts Institute Of Technology, Center For Computational Research In Economics And Management Science: Cambridge, Massachusetts, 1985.
4. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; Hassabis, D. Human-Level Control through Deep Reinforcement Learning. Nature 2015, 518 (7540), 529–533. https://doi.org/10.1038/nature14236.
5. Miikkulainen, R.; Liang, J.; Meyerson, E.; Rawal, A.; Fink, D.; Francon, O.; Raju, B.; Shahrzad, H.; Navruzyan, A.; Duffy, N.; Hodjat, B. Evolving Deep Neural Networks. arXiv:1703.00548 [cs] 2017.
6. Stanley, K. O.; Miikkulainen, R. Evolving Neural Networks through Augmenting Topologies. Evolutionary Computation 2002, 10 (2), 99–127. https://doi.org/10.1162/106365602320169811.
7. Andreas, J.; Rohrbach, M.; Darrell, T.; Klein, D. Neural Module Networks.
8. Minsky, M.; Papert, S. A. Perceptrons an Introduction to Computational Geometry; The Mit Press, 2017.
9. Ivakhnenko, A. G. Polynomial Theory of Complex Systems. IEEE Transactions on Systems, Man, and Cybernetics 1971, SMC-1 (4), 364–378. https://doi.org/10.1109/TSMC.1971.4308320.
10. Lecun, Y. A Theoretical Framework for Back-Propagation. Proceedings of the 1988 Connectionist Models Summer School, CMU, Pittsburg, PA 1988, 21–28.
11. Rumelhart, D. E.; Hinton, G. E.; Williams, R. J. Learning Representations by Back-Propagating Errors. Nature 1986, 323 (6088), 533–536. https://doi.org/10.1038/323533a0.
12. Hochreiter, J. DIPLOMARBEIT IM FACH INFORMATIK Untersuchungen Zu Dynamischen Neuronalen Netzen; 1991.
13. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. Neural Computation 1997, 9(8), 1735–1780.https://doi.org/10.1162/neco.1997.9.8.1735.
14. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE 1998, 86 (11), 2278–2324. https://doi.org/10.1109/5.726791.
15. Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; van den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; Hassabis, D. Mastering the Game of Go with Deep Neural Networks and Tree Search. Nature 2016, 529 (7587), 484–489. https://doi.org/10.1038/nature16961.
16. Duch, Wlodzislaw; Norbert Jankowski. Survey of Neural Transfer Functions.(1999).

17. *Leipzig Corpora Collection:* Tamil community corpus based on material from 2017. Leipzig Corpora Collection. Dataset. https://corpora.uni-leipzig.de?corpusId=tam_community_2017.
18. Divyanshu Kakwani and Anoop Kunchukuttan and Satish Golla and Gokul N.C. and Avik Bhattacharyya and Mitesh M. Khapra and Pratyush Kumar, IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages, Findings of EMNLP (2020)
19. Malarkodi Nathan and Lalitha devi, Fine-Grained Named Entity Recognizer for Tamil, Tamil Internet Conference (TIC 2019)
20. Lakshmanan, BalaSundaraRaman ; Ravindranath, Sanjeeth Kumar, Theedhum Nandrum@Dravidian-CodeMix-FIRE2020: A Sentiment Polarity Classifier for YouTube Comments with Code-switching between Tamil, Malayalam and English, FIRE 2020

கே ஃபோர் கீபோர்டு இரண்டாம் பதிப்பின் புது உத்திகள்

முனைவர் வெ. கிருஷ்ணமூர்த்தி
முன்னாள் பேராசிரியர், அண்ணா பல்கலைக் கழகம்
லேர்ன்ஃபன் சிஸ்டம்ஸ்
prof.vkrish@gmail.com

Key Words
Visually challenged, Keyboard, பார்வைக் குறைபாடு, விசைப்பலகை

கட்டுரைச் சுருக்கம்

கே ஃபோர் கீபோர்டின் முதல் பதிப்பை உருவாக்கும்போது, இந்தப் புதிய தொழில்நுட்பமானது, பார்வைக் குறைபாடு உள்ள இளம் தலைமுறையினருக்கு எல்லா வழிகளிலும் பயன் உள்ளதாக இருக்க வேண்டும் என்பது முதன்மையான நோக்கமாக இருந்தது. அதனால் அதிக அளவில் செயல்பாடுகள் கொடுக்கப்பட்டிருந்தன. இரண்டாவது பதிப்பைத் தொடங்கும்போது, எளிமையாக இருக்க வேண்டும் என்பது முதன்மை நோக்கமாக வைக்கப்பட்டது. இதனால், ஏற்கனவே இருந்த சில செயல்பாடுகள் நீக்கப்பட்டன. சில மாற்றப்பட்டன. சில சேர்க்கப்பட்டன.

ஒரு தடவல் எல்லா உள்ளீடுகளுக்கும் என்னும் கோட்பாடு செயல்படுத்தப்பட்டது. இதனை நிறைவேற்றும் விதத்தில் தடவலின் வரையறை சற்று மாற்றி எளிமையாக்கி வைக்கப்பட்டது. இப்படிச் செய்வதால் ஒரு தடவலுக்கு அனுமானிக்கப்படும் சொற்கள் சில சமயங்களில் ஒன்றுக்கும் மேற்பட்டு வரும் நிலமை உருவாகிறது. இதைக் குறைக்க, உயிர் நீட்சி என்னும் ஒரு புது உத்தி செயல்படுத்தப்பட்டுள்ளது. பல மொழிகளிலும், உயிர்மெய் எழுத்து ஒரு தனி எழுத்தாகவே பார்க்கப்படுகிறது. மெய்யுடன் உயிர் சேர்த்தே பார்க்கப்படுகிறது. இதன் அடிப்படையில் இந்த உத்தி உருவாக்கப்பட்டுள்ளது. இதனால் சுமார் 90 விழுக்காட்டிற்கும் மேலாக, ஒரு தடவலுக்கு நாம் நினைக்கும் சொல்தான் முதலில் வரும். இது உள்ளீட்டு வேகத்தை அதிகரிக்கிறது.

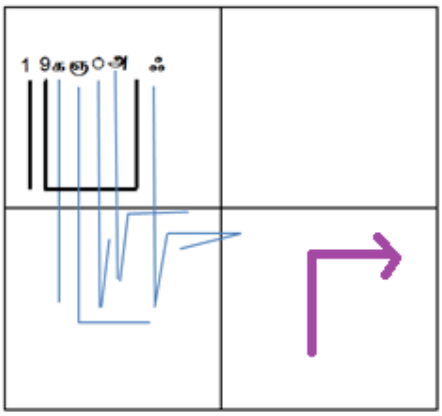
ஒரு தடவலில், தசமப் புள்ளி உட்பட, ஒரு எண் முழுவதையுமே உள்ளிட வகை செய்யப்பட்டுள்ளது. ஒரு தடவலில் ஒரு சொல்தொடரையும் உள்ளிடலாம். ஒரு தடவலில், ஒரு குறியீட்டையும், ஒரு ஈமோஜி அல்லது பல ஈமோஜிக்களையும் கூட உள்ளிடலாம். ஒரு தடவலில் சீர்மைச் செயல்பாட்டையும் செயல்படுத்தலாம். இவை ஒன்றுக்கொன்று மோதிக்கொள்ளாமல் தனித்தனியே செயல்படும்.

தமிழில் ஒரு சொல்லில் பத்துக்கும் மேற்பட்ட நீட்சிகள்கூட இருக்க முடியும். எத்தனை நீட்சிகளுடன் எவ்வளவு பெரிய தமிழ்ச் சொல்லாக இருந்தாலும், ஒரே தடவலில் பெறும் வசதி புதிதாகச் சேர்க்கப்பட்டுள்ளது. இங்கும் வேகத்தை அதிகரிக்க ஒரு புதிய உத்தி பயன்படுத்தப்பட்டுள்ளது. தொடக்கநிலை கணிப்புகள் இந்த விசைப்பலகையானது உள்ளீட்டு வேகத்தை சுமார் இரு மடங்காக்குகிறது என்பதைக் காட்டுகின்றன.

கட்டுரை

முதன்மையான மாற்றங்கள் என்று எடுத்துக்கொண்டால், எல்லாத் தடவல்களும் ஒரே அமைப்பில் செயல்படுவதையும், ஒரு சொல்லைத் தடவும் விதத்தில் செய்யப்பட்ட மாற்றத்தையும் கூறலாம்.

எழுத்துக்கள், சொற்கள், எண்கள், சொற்றொடர்கள், ஈமோஜிக்கள், குறியீடுகள் என்று எல்லாவற்றையும் வெவ்வேறு அமைப்புகளுக்குச் சென்று அந்த அமைப்பில் தடவும் முறை மாற்றப்பட்டு, இவை எல்லாமே ஒரு அமைப்பில் தடவுவதற்கு வகை செய்யப்பட்டுள்ளது. சாதாரணத் தடவல்கள் எண், எழுத்து அல்லது சொல்லைக் கொடுக்கும். இந்த மூன்றுக்குள் வேறுபடுத்த, எழுத்துக்கள் மற்றும் இலக்கங்களின் இடங்கள் மாற்றி அமைக்கப்பட்டுள்ளன. முதல் விசையில், கீழ் நோக்கித் தொடங்கும் இலக்கங்கள் மற்றும் எழுத்துக்களின் கீற்றுகள் கீழே காண்பிக்கப்பட்டுள்ளன. அடிப்படைத் தடவல்களை கீற்று என்போம்.

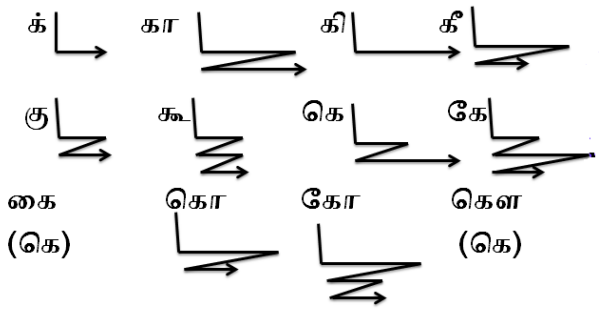


படம் 1. சில கீற்றுகள்

கீற்று தொடங்கும் இடத்தில், அதன் பெயர் கொடுக்கப்பட்டுள்ளது. இலக்கங்களும் எழுத்துக்களும் வெவ்வேறு கீற்றுகளாக இருப்பதைக் கவனிக்கவும். ஒரு கோடு மட்டும் இருந்தால் அது கீற்றாக மட்டும் எடுத்துக்கொள்ளப்படும். எழுத்துக்கள் எல்லாம் ஒரு நீளமான கோட்டுடன் தொடங்குகின்றன. ஆனால், சொல் போன்றவற்றுக்கான தடவல்கள் எல்லாம் ஒரு குட்டையான கோட்டுடன்தான் தொடங்கும். அதனால் எழுத்துக்களுக்கும், இலக்கங்களுக்குமான தடவல்கள் மற்றதற்கான தடவல்களில் இருந்து வேறுபடுகிறது.

ஒரு கீற்றாக இல்லாத தடவலானது, ஒரு சொல், எண், ஒரு சொல்தொடர், ஈமோஜி, அல்லது குறியீடுகளின் தொடர் என்பதில் ஒன்றைக் குறிக்கலாம். தடவலின் கடைசி ஒன்று, இரண்டு அல்லது மூன்று கோடுகள், அந்தத் தடவலின் வகையைக் குறிக்கிறது. இந்த இறுதிக் குறிப்பு எதுவும் இல்லை என்றால் அது சொல்லைக் குறிக்கும். கடைசிக் கோடு ஒரு மூலை விட்டமாக இருந்தால் அது ஒரு எண் சரத்தைக் குறிக்கும். அடுத்த மாற்றம், ஒரு தடவலில் ஒரு எழுத்தை எப்படிக் குறிப்பிடுவது என்பது. முன்பு தொடக்க மற்றும் கடைசி எழுத்துக்கு ஒரு மாதிரியாகவும், இடைப்பட்ட எழுத்துக்களுக்கு வேறு மாதிரியாகவும் இருந்தது. இப்போது எல்லா எழுத்துக்களுக்கும் ஒரே மாதிரியாக மாற்றப்பட்டுள்ளது. ஒரு விசையில் ஒரு எழுத்து இருக்கும் திசையில், ஒரு குட்டையான கோடானது, அந்தத் திசையில் இருக்கும் எல்லா எழுத்துக்களையும் குறிக்கும். இந்த

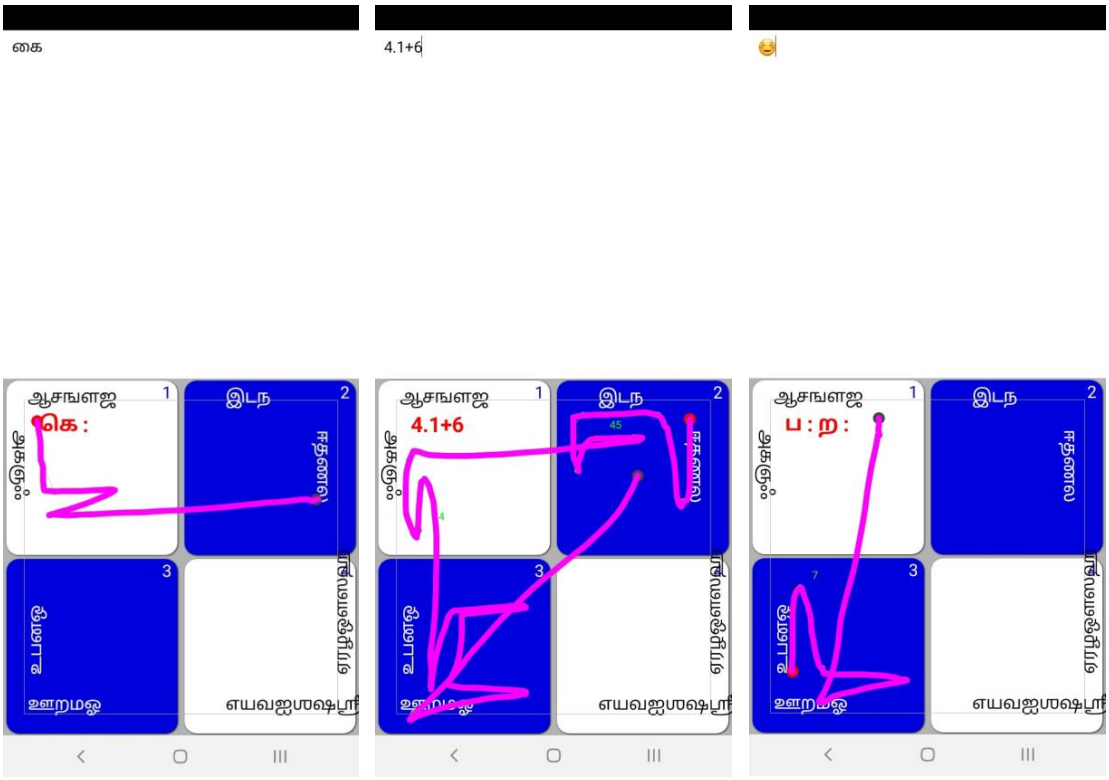
மாற்றம் தடவும்போது சிந்திப்பதை எளிதாக்குகிறது. ஆனால், ஒரு தடவலுக்கு சில சமயங்களில் ஒன்றுக்கு மேற்பட்ட சொற்கள் அனுமானிக்கப்படுவதற்கு வழி செய்துவிடுகிறது. எண் தொடர்களுக்கான தடவல் சரியான ஒரே ஒரு எண் சரத்தைத்தான் வெளியிடும். இங்கு பல அனுமானங்களுக்கு இடமில்லை. இது மற்ற தடவல்களில் இருந்து மாறுபட்டு இருப்பதைக் கவனிக்கவும்.



படம் 1. உயிர் நீட்சிகள்

அடுத்த மாற்றம், அனுமானங்களைக் குறைப்பதற்குப் பெரிதும் பயன்படுகிறது. இந்திய மொழிகள் போன்றவற்றில், உயிர்மெய் எழுத்துக்களை ஒரே எழுத்தாகத்தான் பார்க்கிறோம். மெய்யுடன் உயிர் சேர்வதற்கு, முதலில் மெய் அடுத்து உயிர்க்குறியீடு என்று உள்ளிடுகிறோம். இந்த உயிர்க்குறியீட்டினைத் தனியாக உள்ளிடாமல், மெய்யெழுத்து முடியும் இடத்தில், தொடர்ந்து கொடுப்பது இந்தப் புது முறை. எடுத்துக்காட்டாக, 1, கீழ், வலம், என்னும் தடவல் க் மற்றும் ஞ் என்ற எழுத்துக்களைக் குறிக்கும். மெய்யெழுத்தின் திசைக்கு செங்குத்தாக, விசைப்பலகையின் உட்புறம் நோக்கிய ஒரு குட்டையான கோடு, மெய்ப்புள்ளியைக் கொடுக்கிறது. இந்த செங்குத்துக்கோட்டை உயிர் நீட்சி என்போம். இதேபோல், எல்லா உயிர்க்குறியீடுகளுக்கும் ஒவ்வொரு உயிர் நீட்சி உள்ளது. அவை, குட்டை - மெய்ப்புள்ளி. அடுத்தடுத்த இரு நெட்டைகள் - ஆ. நெட்டை - இ. அடுத்தடுத்த இரு குட்டைகள் - உ. குட்டையும் அடுத்து நெட்டையும் - எ, ஐ, ஒள. ஒரு குறிலுக்கான

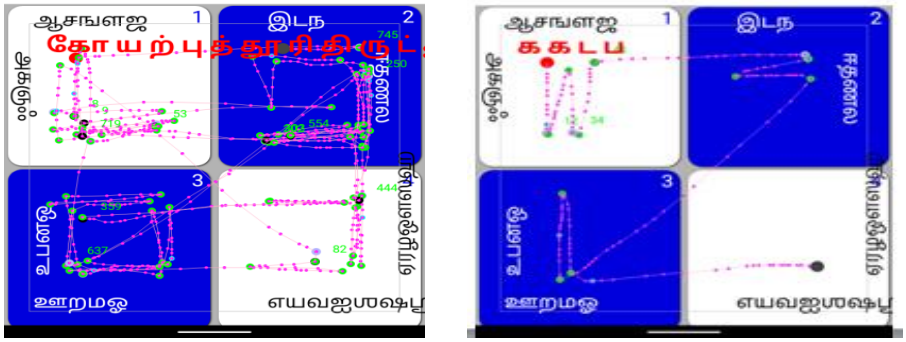
நீட்சியுடன் கடைசியில் ஒரு குட்டையைச் சேர்ப்பது, அந்தக் குறிலுக்கான நெடிலைக் கொடுக்கும். எடுத்துக்காட்டாக, மூன்று குட்டைகள் ஊகாரத்தைக் கொடுக்கும்.



படம் 2. ‘கை’, ‘4.1+6’, ‘ப ம - புன்சிரிப்பு முகம்’ ஈமோஜி

இந்த நீட்சிகள் தமிழ் எழுத்துக்கள் இருக்கும் விழுக்காட்டை அடிப்படையாக வைத்து அமைக்கப்பட்டுள்ளது. ஒரு உயிர் நீட்சி இருந்தால் அதற்கு முன்னால் இருப்பது ஒரு மெய் என்பது தெரியும். அதனால் முன்னால் இருக்கும் குறியீட்டுக்கான அனுமானங்கள் பெரிதும் குறைகின்றன. இது அனுமான நேரத்தைக் குறைக்கிறது. உயிர் நீட்சியானது உயிர்களின் இடத்தைச் சரியாகக் குறிப்பிடுவதால், அனுமானங்களின் எண்ணிக்கை பெரிதும் குறைந்து, நாம் நினைக்கும் சொல் முதலில் வருகிறது. எண்களை உள்ளிடும்போது, இலக்கங்களுடன் புள்ளியும் காற்புள்ளியும் தேவைப்படுகிறது. ஒரு கணிப்பைச் செய்ய கூட்டல், கழித்தல், பெருக்கல், வகுத்தல் குறிகளுடன், பிறைக்குறியீடுகளும் தேவைப்படுகின்றன. இந்த எல்லாக் குறியீடுகளும் இலக்கங்களுக்குப் பக்கத்தில் வைக்கப்பட்டுள்ளன. இவற்றையும் சேர்த்து ஒரே தடவலில்

பெறலாம். எடுத்துக்காட்டாக, 23.45+(67-14). இதை உள்ளிட்டவுடன் ஒரே தடவலில் இதைக் கணக்கிட்டு விடை காணவும் முடியும். கணிப்பானுக்காக விசைப்பலகையை விட்டு வெளியே செல்ல வேண்டியதில்லை. எண்களையும் எழுத்துக்களையும் சேர்த்து உள்ளிடுவது உள்ளீட்டை எளிதாக்குகிறது.



படம் நீளமான சொற்கள். மற்றும் சொற்றொடர்

தென்னிந்திய மொழிகளில் ஒரு சொல்லில் பத்துக்கும் மேற்பட்ட நீட்சிகள்கூட இருக்க முடியும். எத்தனை நீட்சிகளுடன் எவ்வளவு பெரிய தமிழ்ச் சொல்லாக இருந்தாலும், ஒரே தடவலில் பெறும் வசதி புதிதாகச் சேர்க்கப்பட்டுள்ளது. இங்கும் நீளமான சொற்களை உள்ளிடுவதில் நேரத்தைச் சற்றே குறைக்க இன்னொரு உத்தி செயல்படுத்தப்பட்டுள்ளது. தமிழில் மெய்யெழுத்துக்களும், இகர, உகர உயிர்மெய்களும், அதிகம் பயன்படும். பெரிய சொல்லின் நடுவில் வரும் இந்த எழுத்துக்களை, ஒரு நீண்ட செங்குத்துக் கோடு, நீண்ட படுக்கைக் கோடு, மற்றும் நீண்ட மூலைவிட்டக் கோடு இவற்றால் பெறலாம். இவற்றை எந்த விசையில் இருந்தும் செய்யலாம். இங்கு உயிர்க்குறியீடு நிலைப்படுத்தப்படுகிறது.

மெய்யானது எதுவாகவும் இருக்கலாம். என்றாலும், பெரிய சொல்லில் இதன் தாக்கம் அதிகம் இல்லாமல் நாம் நினைக்கும் சொல் முதலில் வந்து விடும்.

ஈமோஜிக்களை அவற்றின் பெயர்களை வைத்து ஒரே தடவலில் உள்ளிடும் முறை, ஆயிரக்கணக்கான ஈமோஜிக்களையும் எளிதாக உள்ளிடுவதை சாத்தியமாக்குகிறது. இதே போல், எல்லாக் கணிதக் குறியீடுகளையும் எளிதாக உள்ளிடுவது மாணவர்களின் எதிர்காலத்திற்குப் பெரிதும் பயன்படும். தொடக்கநிலை கணிப்பு இந்த விசைப்பலகையானது உள்ளீட்டு வேகத்தை சுமார் இரு மடங்காக்குகிறது என்பதைக் காட்டுகின்றது.

முடிவுரை

இந்த விசைப்பலகையில் குறிப்பிடப்பட்டுள்ள கோடுகளின் நீளமும் திசையும் விரும்பத்தக்கவை என்ற அளவில் மட்டுமே இருக்கின்றன. எவராலும் இவற்றை அப்படியே செயல்படுத்த முடியாது என்பதே உண்மை. அதனால், இந்த வரையறைகளில் இருந்து மாறுபட்டிருந்தாலும், விசைப்பலகை வேலை செய்யும்படி செயல்படுத்தப்பட்டுள்ளது. இங்கு கொடுக்கப்பட்டுள்ள எடுத்துக்காட்டுப் படங்களைப் பார்த்தால் இது தெரியும்.

இந்த விசைப்பலகையில் உள்ளிடுவது, சாதாரண விசைப்பலகையில் உள்ளிடும் நேரத்தில் சுமார் பாதி அளவே எடுத்துக்கொள்கிறது. நேரம் மிச்சமாதல், எளிதான இயக்கம், எண்கள் மற்றும் குறியீடுகளைக் கையாளும் முறை என்று பல விதங்களிலும் சாதாரண விசைப்பலகைகளை விட கே ஃபோர் கீபோர்டு சிறப்பாகச் செயல்படும்படி இந்த இரண்டாம் பதிப்பு வடிவமைக்கப்பட்டுள்ளது.

Tamil News Classification using LSTM

M. Pavithrah
Dept. of Information Technology
Chennai Institute of Technology
Email: pavithrahmit2019@citchennai.net

S. Nandhini
Dept. of Information Technology
Chennai Institute of Technology
Email:nandhinisit2019@citchennai.net

R. Janarthanan
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: janarthananr@citchennai.net

R. Ponnusamy
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ponnusamyr@citchennai.net

Abstract

Tamil Newspaper Classification is the one important and unique task performed over the information systems. Attaining a high accuracy for Tamil Text Classification is a challenging task. It is proved that deep learning proved to be effective in doing different complex text information processing and classification. Most of the existing classification systems of Tamil Information Processing Systems use the only conventional model. Therefore, in this work, an attempt is made to classify the text using deep learning algorithms. In this work, an attempt is made to classify the Tamil News Paper text using Long Short Term Memory (LSTM) System. The nouns, verbs are extracted and the term-frequency, inverse document frequency are taken from the training and testing documents by eliminating the stop words. This input is given to the LSTM machine to extract important features which enable the system to classify the text. It is performing a multiclass classification with few classes. At first, the system uses the different Tamilmurasu Newspaper Dataset topics collected from the Kaggle repository for both training and testing. Standard measurements like accuracy, precision, recall, and F1-measures are calculated and presented.

Keywords: Newspaper Classification, Long Short Term Memory (LSTM) Tami Text, Accuracy, Precision, Recall, F1-measure.

1. Introduction

Text classification is one of the significant tasks in text mining and information retrieval. It is also a challenging task while doing it for the Tamil Language. Tamil text is a non-roman letter script and therefore information processing needs additional steps to do any normal English-like Language Processing. It means that it is a multi-glyph-based information processing system compared to English-like languages. Text

Classification is the task of finding and assigning the given text into different classes. In a real-time environment text analysis, organizing, sorting text data set into different classes is a time-consuming task.

In the existing classification system, most of the researchers use only conventional methods, like Naive Bayes, Decision Tree, K-Nearest Neighbor, Support Vector Machine, etc. Accuracy and the semantic validity of the classified documents is the basic measure to decide the quality of the classification system. In this case of conventional methods, it is questionable. The long short-term memory (LSTM) model for text classification produces accurate results and has been recently used in various Information Retrieval process. Therefore, in recent times it is proved that Deep Learning algorithms played a vital role in information mining. In this work, we attempt to classify the Tamil Text using the LSTM Machine.

The Long Short-Term Memory (LSTM) system is a recurrent neural network that can learn the order dependence of classification problems. The Tamil text classification process is divided into three stages:

1. Preprocessing stage: This stage includes stop word removal, stemming, punctuation and special symbol removal.

2. Feature extraction: It consists of statistical methods and language methods, and extracts relevant features from documents for classification.

3. Processing stage: The final stage of Tamil text classification. The text classification system is applied to the extracted features to classify documents.

This paper attempted to classify the Tamil text documents into 15 different predefined categories using Tamilmurasu dataset collected from the Internet. The performance of the system is measured through the standard normal text classification metrics. In this paper, Section 2 reviews the literature, Section 3 describes basic News Categories of linguistic knowledge. Section 4 System Architecture Design in detail. Section 5 provides an overview experimental setup in section, 6 which explains the result and discussion. Section 7 Conclusion about the interpretation of the paper.

2. Background Literature

The Tamil text classification has an important tool for different mining and information retrievals systems. A vast amount of work has been carried out to classify the text worldwide by considering its importance using different algorithms and modern tools. Several data sets are available on the web, leading to the development of Tamil text Data Mining. This section deliberates works related to text classification systems development.

1. Ramraj, S., et al., in 2020 [1] attempted to classify the Tamil news article using CNN models. It is inferred that Convolutional Neural Networks with pre-trained embedding's for the Tamil language gives better results compared to Support Vector Machine and Naive Bayes trained with term frequency and inverse document frequency feature vectors. The precision, recall, and F1 score for the class politics is low when compared to the other two methods.

2. Wang, D., Bai, Y., & Hamblin, D. in 2019 [2], developed an algorithm to spontaneously retrieve critical information from raw data files in National Aeronautics and Space Administration's airborne measurement data archive.

3. Ade Clinton Sitepu, Wanayumini Wanayumini, Zakarias Situmorang in 2021[3] attempted to classify the social media text into bullying and non-bullying using Naïve Bayes Method and reported that 88% of results.

4. Huaiguang Wu, Daiyi Li, and Ming Cheng, in 2019[4] classified the larger Chinese text using SVM and CNN methods. Text classification is performed using SVM multiple classifiers. Experimental results show that the CSVM algorithm is more effective than other traditional Chinese text classification algorithms.

5. D. M. Harikrishna, K. Sreenivasa Rao in 2019 [5] classified stories of Indian languages using Conventional methods, like KNN and SVM. It is observed that the performance of the classifier is evaluated using 10-fold cross-validation, and the effectiveness of the classifier is measured using accuracy, recall, and F metric. It can be seen from the classification results that adding language information can improve the performance of story classification.

6. Winda Kurnia Sari, Dian Palupi Rini, Reza Firsandaya Malik in 2020[6] by changing the boundaries and contrasting the eight proposed LSTM models with enormous scope informational indexes, the confirmations are characterized to show that the LSTM with GloVe elements can accomplish great execution in text grouping. For this situation, the creator additionally detailed 95% or more outcomes.

7. Winda Kurnia Sari; Dian Palupi Rini; Reza Firsandaya Malik, in 2021 [7] attempt to classify the documents classify the web of science documents into seven different subject categories. In this classification, the authors reported 82.13% of results in their experiments.

8. Dennis Dang, Fabio Di Troia, Mark Stamp [8] have attempted to classify the malware using LSTM employ techniques used in natural language processing (NLP), including word embedding and bi-direction LSTMs (BiLSTM), and this work also use convolutional neural networks (CNN). It gives the best classification.

To exploit the LSTM method in this work an attempt is made to classify the Tamil documents using Tamilmurasu dataset [10], which is openly available in the Kaggle repository.

3. Classification Categories

Tamil Murasu [9] is a daily evening Tamil newspaper. It is being published in various parts of Tamilnadu from 1935 onwards. It classifies the news article into 15 Categories and the categories are shown in the following hierarchical tree. In this process of classification, the given document is classified into one of the following categories.

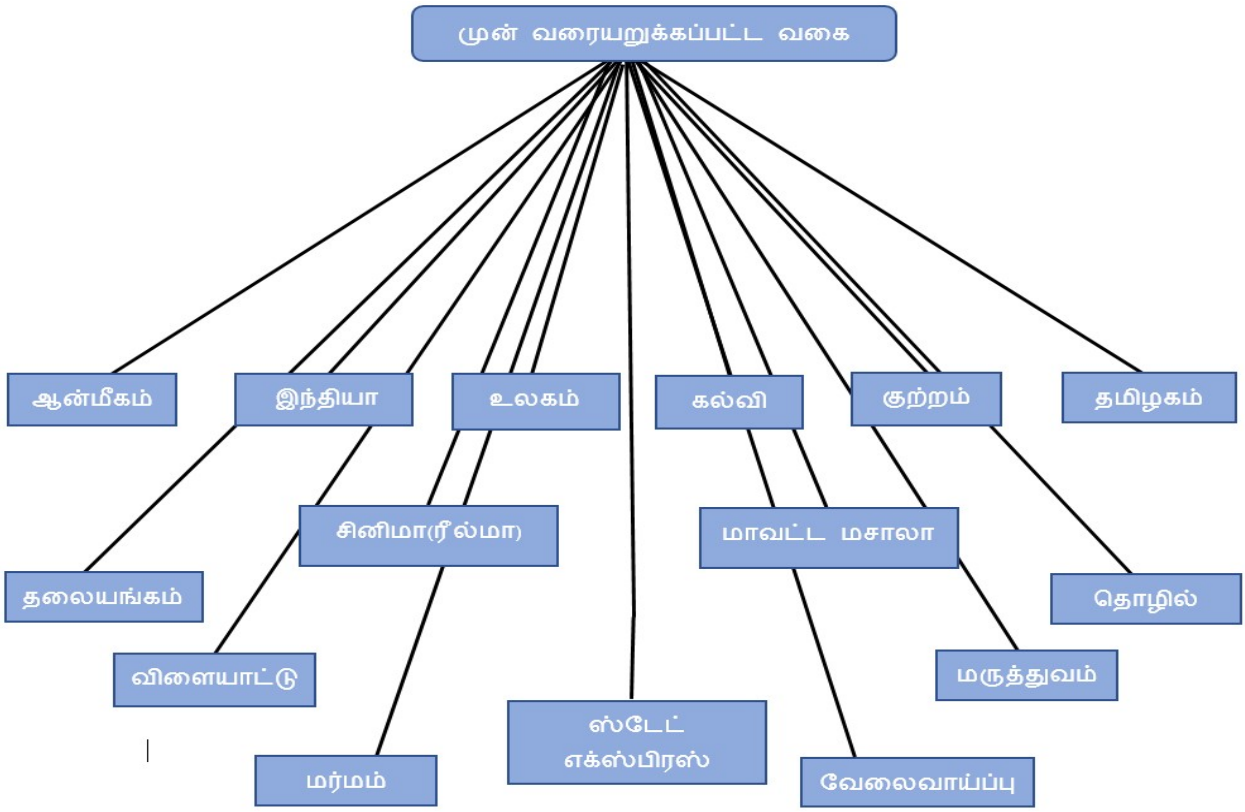


Figure 1 Tamilmurasu Classification Tree

4. System Design

In the arranged model first, the Input of each report is assessed to distinguish the classes. The info variable anticipated through the LSTM model is the basic factor of the Tamilmurasu data set collection. Further LSTM strategy is applied to discover comparative classifications, which has a high likelihood of discovering classes. To show the utility of the proposed approach brief marks of numerical model and methodology are given by X_1, X_2, X_3……X_i is the Document categories predicted in C_1, C_2, C_3….C_i. The proposed system architecture model of the developed system is given in figure 2.

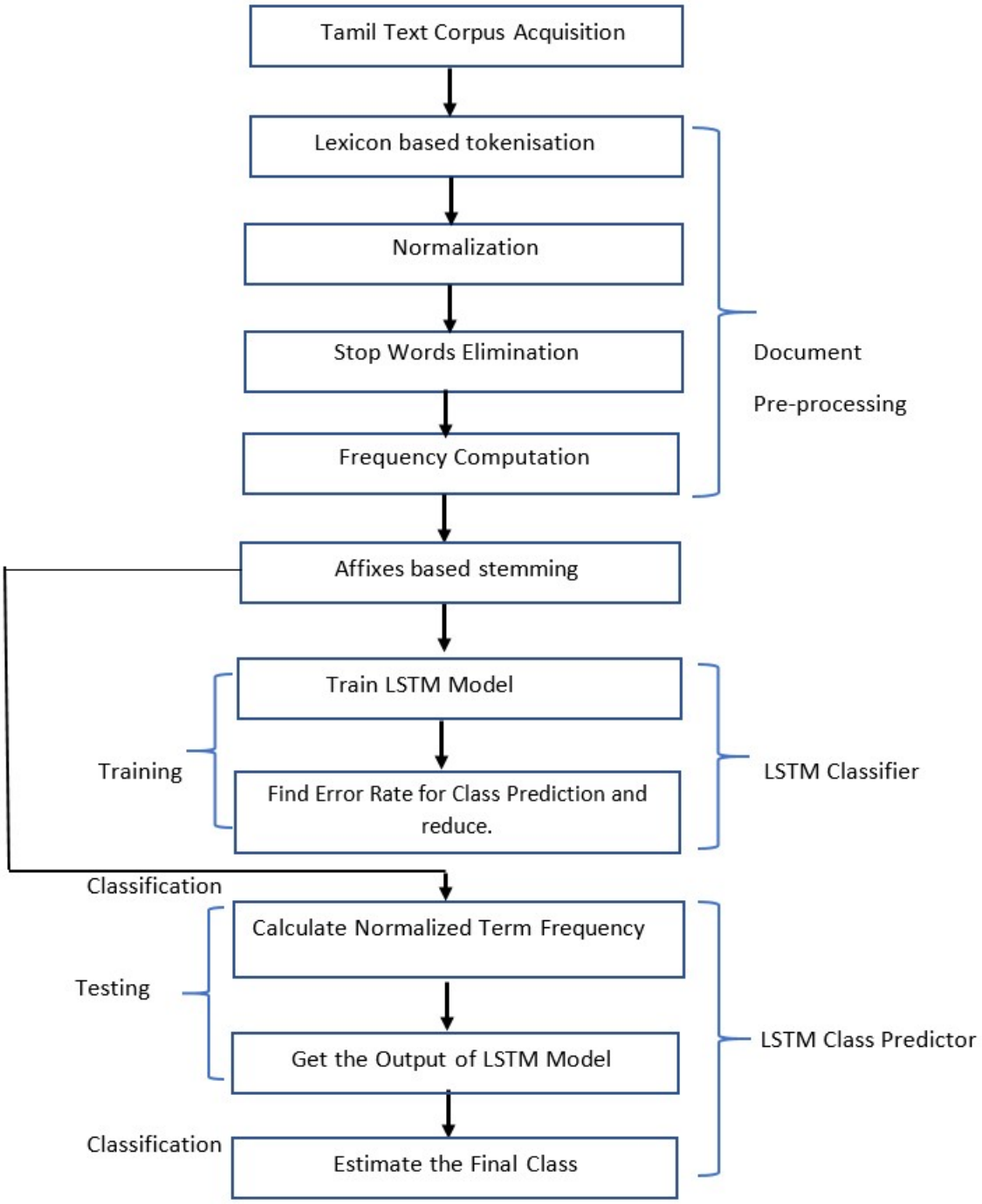


Figure 2. The architecture of Tamil Text Classification System using LSTM Model

Text Corpus Acquisition

A lot of dataset is generally expected to get great order exactness of any AI framework. For that reason a text corpus of Tamilmurasu data set gathered from various Kaggle Repository, is utilized. The corpus was physically characterized into fifteen unique areas specifically news, Tamilnadu, Indian, Crime, Cinema, District, sports, world, state express, heading news, employment, medical, spiritual, education, business, and mystery.

Tokenization

Tokens are taken from the corpus by removing white spaces and punctuation marks. Sometimes the tokens are multiple words. To get the token initially the lexicon is searched, if it is there then it will be taken for counting. The 'tokenization lexicon' is manually prepared and gathered from different sources containing 356,791 unique entries.

Normalization

Given token is converted as the Tamil Unicode text and this will make the system universalize the document. Sometime it contains Arabic alphabets or non-standard word class and are defined through the regular expression. These words and alphabets are standardized and converted into equivalent Tamil Words and strings.

Stop word Elimination

Those words are called as functional words which has no impact on text classification and therefore it should be eliminated. These words are sometimes frequently occurring words and a bag of words list is maintained. It will be eliminated from the list of tokens before proceeding to the next step. In this 219 such words are eliminated from the document text list.

LSTM Model

Long Short-Term Memory (LSTM) networks are an altered adaptation of repetitive neural networks that streamline reviewing data from an earlier time. Here, the RNN's evaporating inclination issue is solved. LSTM is appropriate to distinguish, investigate, and foresee time models given delays of the dubious term. Back-spread is utilized to prepare the model. Three entryways are available in an LSTM organization and it is displayed in Figure 3.

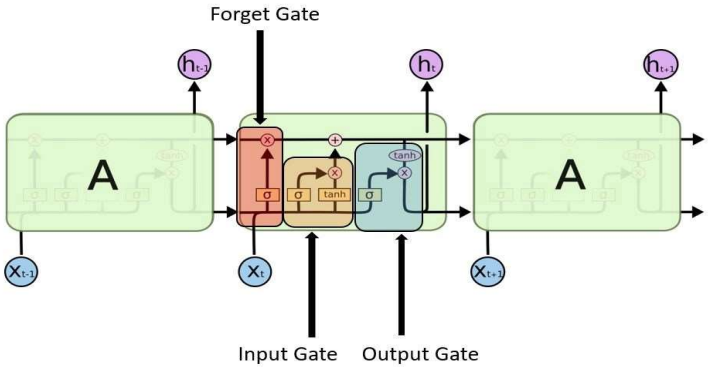


Figure 3 LSTM Gates

1. **Input gate** — Find the value in the entry you want to use to change the memory. The sigmoid function determines the value that passes 0.1. The tanh function weights the sent value to choose its importance between -1 and 1.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Input gate

2. **Forget gate** — Find what you are removing from the block. The sigmoid function determines this. It outputs a value of 0 (ignore this) to 1 (leave as is) for each number in the cell state Ct-first to examine the previous state (ht-1) and the content entry (Xt).

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Forget gate

3. **Output gate** — Input and block memory are used to define the output. The sigmoid function determines values greater than 0.1. The tanh function weights the transmitted values to assess their importance in the range -1 to 1 and multiplies the output of the sigmoid.

$$o_t = \sigma\left(W_o \; [\, h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Output gate

A recurrent neural network is a type of long short term memory. In the current action, the RNN output from the previous step is used as Input. Hochreiter & Schmidhuber created the LSTM. It addressed the issue of RNN long-term dependency, in which the RNN cannot predict words stored in long-term memory but can make more accurate predictions based on current data. RNN does not deliver efficient performance as the gap length rises. By default, the LSTM can keep the information for a long time. It's utilized for time-series data processing, prediction, and classification. The LSTM features a chain structure with four neural networks and various memory blocks known as cells.

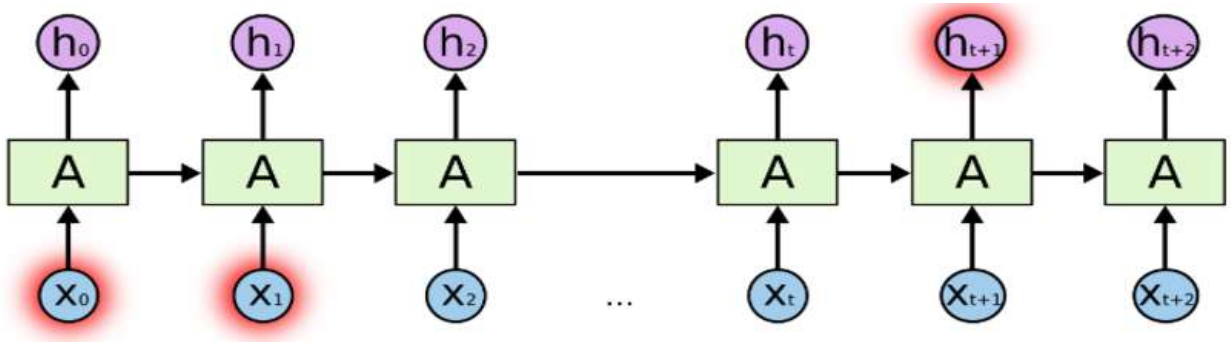


Figure 4 LSTM Chain Organization Model

Finally, when we have numerous yields, softmax function changes over yields layers into a likelihood dissemination. We have 15 marks altogether, but since we didn't one-hot encode names, we need to utilize sparse_categorical_crossentropy as classificaiton work, it assumes 0 is a potential name too, while the tokenizer object which tokenizes beginning with whole number 1, rather than number 0. Accordingly, the last Dense layer needs yields for categories 0, 1, 2, 3, 4, 5… 15 and the number 0 has never been utilized.

5. Experimental

The dataset used for classification is received from the Kaggle free open source repository. It is a Tamilmurasu News Dataset having 127725 News Articles of fifteen different categories. This dataset includes four different attributes, such as News Id, News Category, News Title, and News. In this dataset, the given dataset is divided into two groups. One group for training and another one group for test data. In case of training all attributes data items are taken and in case of testing the News Title and News Article alone take for experimentation.

5.1 Experiment Setup

A python program is developed with the Keras, nltk, Scikit-learn, Pandas, NumPy, Tensorflow packages to model the LSTM model to predict the predefined text categories. This prototype works well for state-of-the-art data in the database. In this program, it will be able to expect only the LSTM model alone. The system is configured to run in a standalone PC Environment.

5. Results and Discussion

The performance of the Tamil text classification system can be evaluated by using four different standard metrics are Accuracy, Precision, Recall, and F1 measure. Precision measures the factualness of the classifier system. Higher precision means fewer false positives, while a lower precision means false positives.

$$\text{Precision} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of extracted text}}$$

Recall measures the completeness, or sensitivity, of a classifier. Higher recall means fewer false negatives, while lower recall means more false negatives.

$$\text{Recall} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of annotated text}}$$

Accuracy measures the overall degree to which instances have been correctly classified,

$$\text{Accuracy} = \frac{Number\ of\ correctly\ classified\ instances}{\text{Total Number of instances}}$$

A weighted harmonic measure mean of precision and recall is F1-measure, which is the rate of a system with one unique rating.

$$\text{F1} - \text{Mesure} = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 1 Performance of Results Evaluation of LSTM System Testing Process

Label	Precision	Recall	F1-Score	Accuracy
1	98	99	99	99
2	97	98	79	98
3	99	99	97	99
4	98	99	96	99
5	97	98	99	98
6	97	98	97	98
7	99	99	99	99
8	98	99	99	99
9	98	99	99	99
10	98	99	99	99
11	99	99	99	99
12	97	98	97	98
13	98	99	99	99
14	96	97	93	97
15	99	99	99	99
Average	97.86	98.66	96.66	98.64

6. Conclusion

Tamil Text Classification is one of the essential tasks in Text Mining and Natural Language Processing. Developing and acquiring more accurate results for Text Classification is a challenging task. This paper proposes a novel LSTM model for Tamil Text Classification into fifteen pre-defined categories. In this method the classification accuracy is 98 percent and above which is comparatively good as compared to the traditional classifiers.

7. References

1. Ramraj, S., Arthi, R., Murugan, S., & Julie, M. (2020). Topic categorization of Tamil news articles using PreTrained Word2Vec embeddings with Convolutional neural network. 2020 International Conference on Computational Intelligence for Smart Power System and Sustainable Energy (CISPSSE). https://doi.org/10.1109/cispsse49931.2020.9212248.
2. Wang, D., Bai, Y., & Hamblin, D. (2019). A machine learning algorithm in automated text categorization of legacy archives. 8th International Conference on Soft Computing, Artificial Intelligence and Applications. https://doi.org/10.5121/csit.2019.90701.
3. Sitepu, A. C., Wanayumini, W., & Situmorang, Z. (2021). Determining bullying text classification using naive Bayes classification on social media. Jurnal Varian, 4(2), 133-140. https://doi.org/10.30812/varian.v4i2.1086.
4. Wu, H., Cheng, M., & Li, D. (2019). Chinese text classification based on character-level CNN and SVM. International Journal of Intelligent Information and Database Systems, 12(3), 212. https://doi.org/10.1504/ijiids.2019.10024507.
5. Harikrishna, D. M., & Rao, K. S. (2020). Children's story classification in Indian languages using linguistic and keyword-based features. ACM Transactions on Asian and Low-Resource Language Information Processing, 19(2), 1-22. https://doi.org/10.1145/3342356.

6. Sari, W. K., Rini, D. P., & Malik, R. F. (2020). Text classification using long short-term memory with glove features. Jurnal Ilmiah Teknik Elektro Komputer dan Informatika, 5(2), 85. https://doi.org/10.26555/jiteki.v5i2.15021.
7. Sari, W. K., Rini, D. P., & Malik, R. F. (2019). Text classification using long short-term memory. 2019 International Conference on Electrical Engineering and Computer Science (ICECOS). https://doi.org/10.1109/icecos47637.2019.8984558.
8. Dang, D., Di Troia, F., & Stamp, M. (2021). Malware classification using long short-term memory models. Proceedings of the 7th International Conference on Information Systems Security and Privacy. https://doi.org/10.5220/0010378007430752.
9. https://www.kaggle.com/vijayabhaskar96/tamil-news-classification-dataset-tamilmurasu
10. www.tamilmurasu.org.

A Conversational AI: LSTM Based Tamil Chatbot system for Academic Information System

P. Karthik
Dept. of Information Technology
Chennai Institute of Technology
Email: karthikpit2019@citchennai.net

Kishore Kumar.L
Dept. of Information Technology
Chennai Institute of Technology
Email:kishorekumarlit2019@citchennai.net

Venkateswar. S
Dept. of Information Technology
Chennai Institute of Technology
Email:venkateswarsit2019@citchennai.net

B. Sundarambal
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: sundarambalb@citchennai.net

R. Ponnusamy
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ponnusamyr@citchennai.net

Abstract

A Chabot is a system used to conduct an online conversation via text to speech and give the impression of actual human conversation. It is designed to simulate the system like human communication. The system should automatically answer all the queries raised by the user. Nowadays, the number of queries raised by aspiring students in the academic system increasing dramatically. It is very hard for the user to answer the queries raised by the aspiring students about the various courses offered by the academic institution. It is good to have an automated chatbot system to answer the queries raised by the students. To solve this problem a chatbot system is designed using Conversational AI. Specifically, the design of LSTM based system to order the sequence of words spoken by the system is effective. In addition to that usage of the NLP, toolkit makes the system more meaningful. Tamil has to follow a strict grammar rule. If these words are rearranged, the original meaning is lost. Thus, it is hard for processing the data for the computers when the original meaning is lost. Thus, one requires to train the models with specific structures and discard others. Therefore, in this work, an attempt is made to design an LSTM/Deep Learning-based system ordering the sentence is effective. It is designed to measure the user satisfaction rate and evaluation rate to ensure the effectiveness of the system.

Keywords: Chatbot, Conversational AI, Deep Learning, LSTM, user satisfaction, evaluation rate.

1. Introduction

In the open economy service offered to the people are increasing day by day. Most of the companies like to have an IVRS (Interactive Voice Response System) because of the growing demand for products and services. This IVRS needs lots of interaction from the customer. Therefore, the customers avoid such kind of system. Alternatively, it is impossible to provide 24X7 customer services to enquire made by the customer. Therefore it is essential to find an alternative mechanism to solve this issue, which is more interactive and should provide all required information in understanding user requirements.

Artificial Intelligence and Deep Learning is modern emerging technology providing various types of powerful solutions to these kinds of problems. Conversational AI is a set of technologies behind automated messaging and voice-enabled applications that enable human-like interactions between computers and humans. Conversational AI [9] can communicate like a human by recognizing languages and texts, understanding intent, decoding different languages, and responding in a way that mimics human conversation increase. In particular, it can work in a variety of languages.

Several attempts were made by the early researchers to design and develop Tamil Chabot to provide an interactive system to answer the queries in Tamil. All the existing systems are developed with a rule-based system or database-oriented one. Most of the time it will be not able to answer the new type of queries raised by the user. In this system, an attempt is made to develop a dynamic Chatbot with Conversational AI using the Rasa tool, which is capable of dynamically accepting different kinds of queries in Tamil. It is one of the most advanced developments in the Chabot environment. Chabot is an application that allows you to start and continue conversations using auditory and/or textual methods, much like humans. Chatbots are either simple rule-based engines or intelligent applications that use natural language understanding. Today, many companies are beginning to use chatbots extensively. Chatbots are becoming popular because they are available 24 hours a day, 7 days a week, provide a consistent customer experience, serve multiple customers at the same time, and are inexpensive to improve the overall customer experience.

In this paper, Section 2 reviews the literature, Section 3 describes the basics of Chatbot system and System Architecture Design in detail. Section 4 provides an overview experimental setup in section, 5 which explains the result and discussion. Section 6 concludes the paper with the study of the impact of the Tamil Chabot system.

2. Background Literature

The Tamil chatbot is an important tool for communicating with the system to enable the user to get any information from the system in all 24 x 7 hours. A vast amount of work has been carried out to classify the text worldwide by considering its importance using different modern

tools. Several frameworks and algorithms are available on the web, leading to the development of Tamil Chatbot. This section deliberates works related to Chatbot systems development.

1. T. Kalaiyarasi, et al., in 2003 [1] developed a chatbot called Poongkuzhali that communicates in Tamil on a basis of giving an appropriate response to the given question or statement as input using artificial intelligence. The response will be given even when the question phrase is not given incorrect phrase using a set of decomposition rules and a set of reassembly rules in the knowledge base.
2. Raphael Meyer von Wolff, et al., in 2019 [2] attempt to conduct a structured literature review on research papers based on chatbots and says their current potentials, their objectives, the gap present in the research and to tackle them.
3. B.Galitsky, in 2019 [3] explains the use of Explainable ML and its features in the field of chatbot. It also show's a problem and its solution by using the transparent rule-based or ML method.
4. Daniil Sorokin, et al., in 2018 [4] approaches the entity linking in the context of question answering task and jointly optimized neural architecture for entity mention detection and entity disambiguation that models the surrounding content on a different level of granularity. The Wikipedia knowledge base is used as a dataset for question answering data training.
5. Hamid Zafar, et al., in 2018 [5] studies the question answering system over knowledge graphs which map an input into queries. By using Tree-LSTM, it exploits the syntactical structure of input questions as well as candidate formal queries to compute the similarities.
6. Senthilkumar, M., & Chowdhary, C. L. in 2019 [6] developed a sequence to sequence neural network model human to machine chatbot to replace the human. It is very productive compared to other normal chatbots.
7. Jungwook Rhim, et. al., in 2021 [7] surveyed different kinds of Chatbots and used three factors for surveying, such as respondents' perceptions of chatbots, interaction experience, and data quality. It also explained a different kind of Chatbot used in the environment.
8. Prissadang Suta, et. al., in 2021[8] developed a machine learning-based chatbot which handles three steps effectively, understanding the natural language input; generating an automatic, relevant response; and, constructing realistic and fluent natural language responses. The main issue of the natural language understanding problem is solved in this method.

3. RASA System Architecture

The following figure gives an overview of the Rasa open-source architecture. It is a readymade framework available in python for effective chatbot deveoopment and it is a machine learning framework. It has several components such as an action server, tracker store, lock store, file system, dialog policy, NLU pipeline, agent, and so on. The two main components are Natural Language Understanding (NLU) and Dialog Management. NLU is the part that handles intent classification, entity extraction, and response retrieval. It is shown below as the NLU pipeline because it uses the NLU model generated by the trained pipeline to handle the user's utterances.

The dialog management component determines the next action in the conversation, depending on the context.

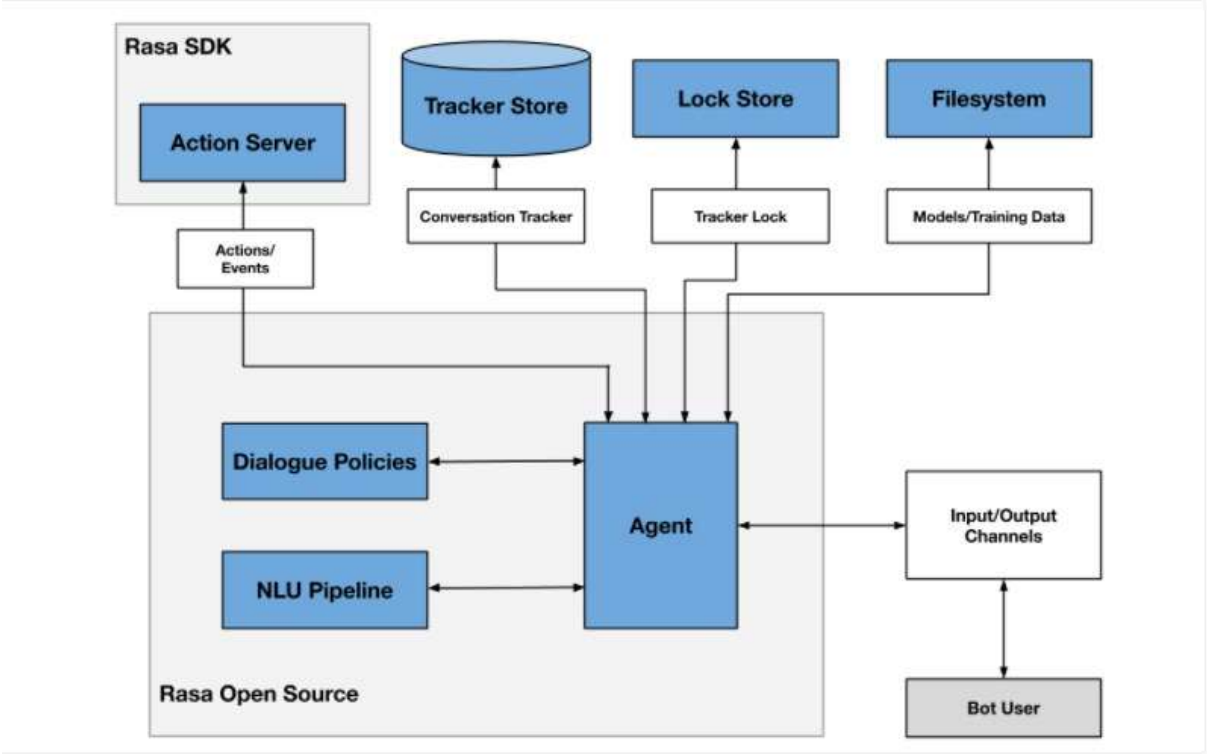


Figure 1. RASA Architecture

Action Server:

Custom actions in the rasa project run using this action server. When an action is predicted using rasa, then the rasa server sends a POST request in the form of JSON payload. When the server finishes, it returns the responses and events in the form of a JSON payload. A conversion tracker is created for the user's responses and adding the events. A webhook endpoint is accepted for HTTP POST requests in the case of using other action servers.

Track Store:

The conversations made in rasa are stored in a place called tracker stores. This open-source tool helps to create custom storage boxes. The default track store is In Memory Tracker Sore. Databases that can be used in rasa are PostgreSQL, Oracle, and SQLite. Mongo Tracker Store and Dynamo TrackerStore are also used to store the history of conversation in rasa.

Lock Store:

Lock store is used in rasa to ensure whether the incoming messages are processed in the correct order, then they are directly stored in lock stores for active processing. Multiple servers are used

and no need for conversation sent by the client. The default lock store in rasa is In Memory LockStore. If the need for a persistence layer in conversation locks, RedisLockStore can be used.

File system:

Rasa collects and loads the training data and this data is imported using a custom importer. RasaFileImporters and MultiProjectImporter are used for importing the data. Then these data are stored in the File system.

Dialogue Policies:
Machine-learning and rule-based policies are used by rasa to decide which action to take place in a conversation. Actions can be selected on each configuration rasa takes, priority for each policy can also be given to make it higher or lower case. Machine learning policies are the TED policy, UnexpecTED Intent policy, Memoization policy, and Augmented Memoization policy. Rule-based policies are Rule policy and Configuring policies.

NLU Pipeline

The components of the NLU pipeline are Tokenizers, Features, Intent Classifiers, and Entity Extractors. Tokenizers will split the text into a text known as tokens. Featured is used for generating the numeric features in rasa. Two types of features like sparse and dense features are used. In rasa Intent Classification, the DIET algorithm is used. Entities don't need an algorithm to detect, so RegexEntityExtractor is used. Thus in NLU, actions are predicted, thus an NLU pipeline can be developed.

Agent

An agent in rasa will help us to train a model, load and will help us to use it. In simple words, it's a simple API that will help us to access Rasa's core functionality.

Bot User.

After the creation of input and output channels, that is when a complete rasa project is developed, a bot user will be created, and thus it helps us to solve the questions asked by humans.

Rasa Framework Layout: Basically, Rasa has the following built-in modules that are used when implementing Chabot:

1. RASANLU for understanding user messages. This part of the framework is a tool/library for intent classification and entity extraction from query text. Entities and intents also allow for the retrieval of responses and the composition of spoken text. You can use this component alone to create a simple and minimal AI chatbot yourself. This is usually the case when creating an AI chatbot that answers FAQs, simple queries, and so on. This blog also uses it to code chatbots.
2. Have a conversation with RASA Core and decide what to do next. This component is the framework's dialog engine that helps build more complex AI assistants that can handle

context (previous requests and responses in a conversation) during a response. RASA recommends using both NLU and Core, but these can be used separately.

3. Integration with Deployment Channels: These components allow you to connect and deploy your bot to popular messaging platforms. Learn more about this here. These help developers focus on bot functionality rather than the installation required for a real deployment. RASA X is a toolset that takes bots (developed with RASA open source) to the next level. It's free, but it's a closed toolset available to all developers. The following figure illustrates how the Rasa chatbot works.

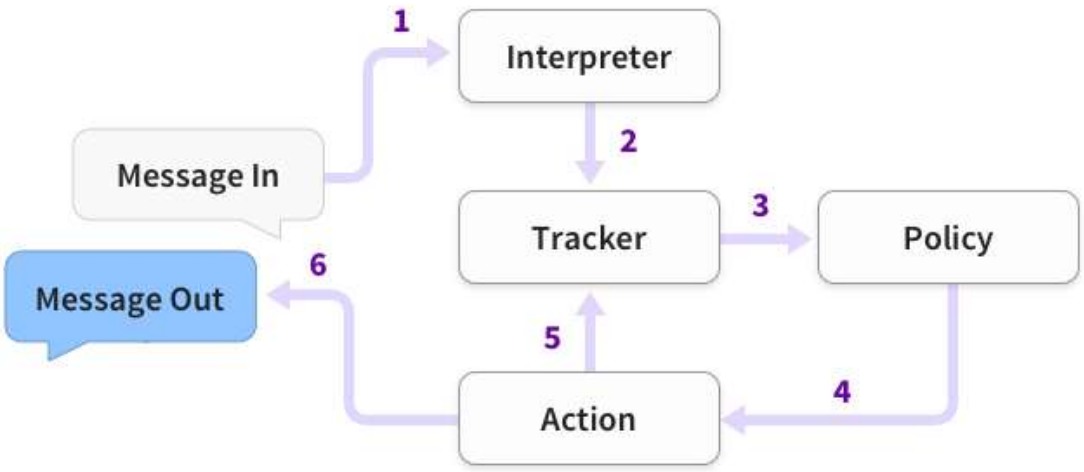


Figure 2 Method of working of Rasa Chatbot

The following steps are described below.

1. First, the message is received from the user in the bot and passed to the interpreter. The interpreter translates the message into a dictionary containing the original text, intents, and all found entities. This part is handled by NLU.
2. Next, the tracker is an object that tracks conversation status. You will receive information that a new message has arrived. It also helps you keep track of the entities extracted by the user and how the conversation is managed. NS. Dialogue flow.
3. The policy receives the current status of the tracker.
4. The policy determines the next action to take.
5. The selected action is logged by the tracker to help you follow the path or flow of the conversation.
6. The response is sent to the user. The user then responds to the response received from the bot.

4. Experimental

Intents are developed in Tamil to make the system customize the Tamil language. Responses were also created for the intents. Using these intents and responses Rasa trains data in the Tamil language. A set of stories are created which explain the situation sequence of activities to perform in the system. Rules are created to adapt the queries of different formats. In this system, nearly 46 intents and 140 responses are created in Tamil. Also, 20 stories and rules are created for intents and responses.

4.1 Experiment Setup

A python program is developed with an open-source machine learning framework is called the Rasa tool used for this system development. This uses python dependency packages like tensor flow, Spacy. It is configured to run through the GUI or Command prompt. In the experiment, the system is simulated using a PC environment connected to the Internet.

4.2 Simulation Screen

The Simulated system is configured to work for the Chennai Institute of Technology for admission seekers on the campus. This Chatbot system enables the user to chat with the system in Tamil and its screen snapshots are shown in the following figure 3.

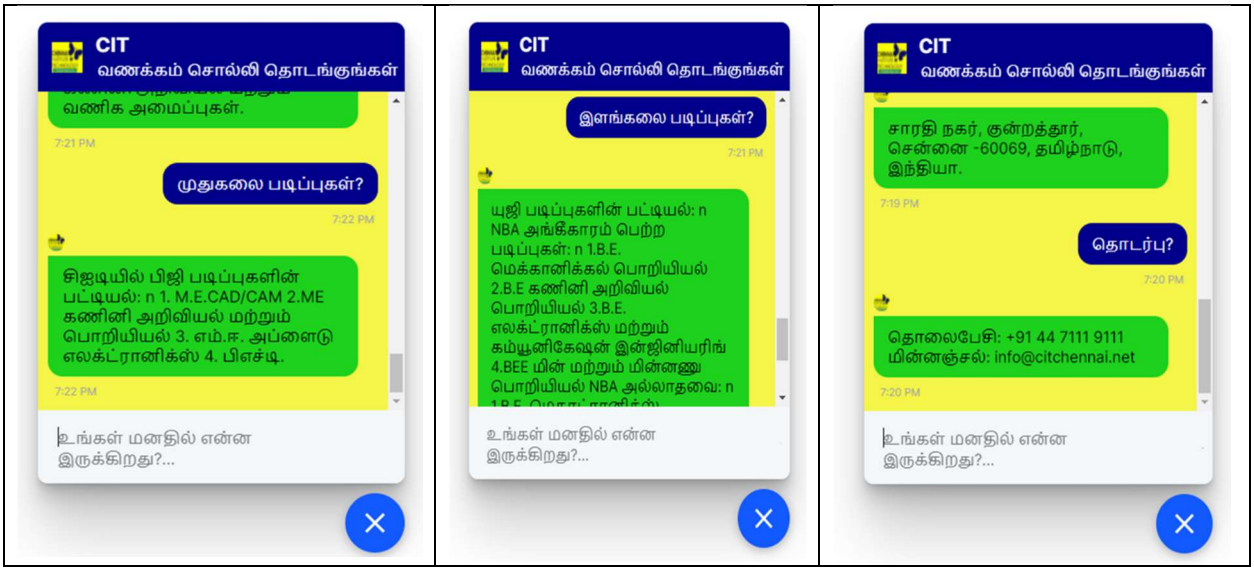


Figure 3 Chatbot screen snapshots

5. Experimental Results and Discussion

In general, there is different kinds of measurements [10] are used for measuring the value of the Chatbot systems [9]. These attributes are 1. User Satisfaction 2. Dialog Efficiency Metric and 3. Several Queries and Response, etc. In this system, explicit feedback is collected from the user to get their user satisfaction level and satisfaction level is plotted as a bar chart shown in Figure 4. Secondly, the efficiency is measured of 8 sample dialogues in terms of the atomic match, First-word match, Most significant match, and No match. To measure the efficiency of the adopted learning mechanisms to see if they increase the ability to find answers to general user input as shown in Figure 5.

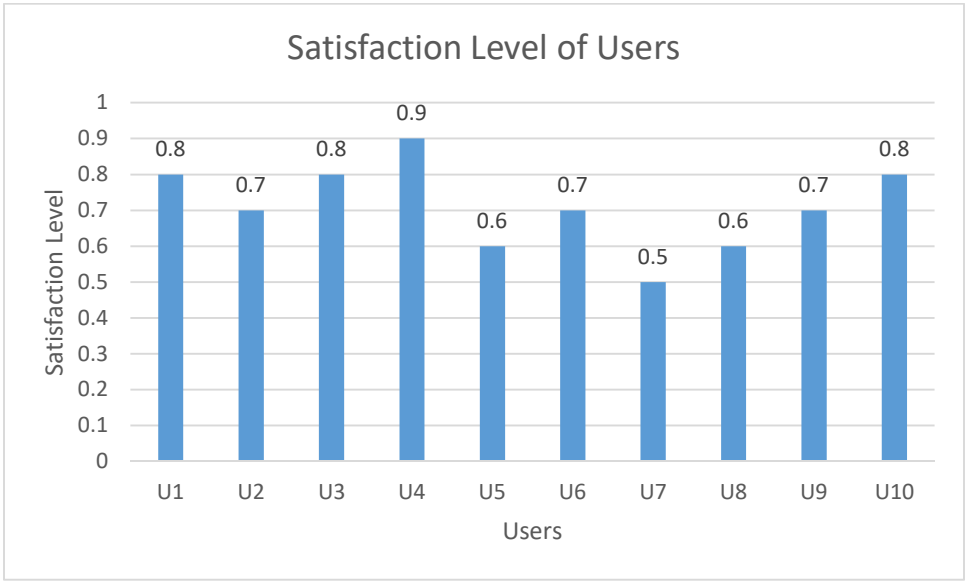


Figure 4 – User Satisfaction Level Measures.

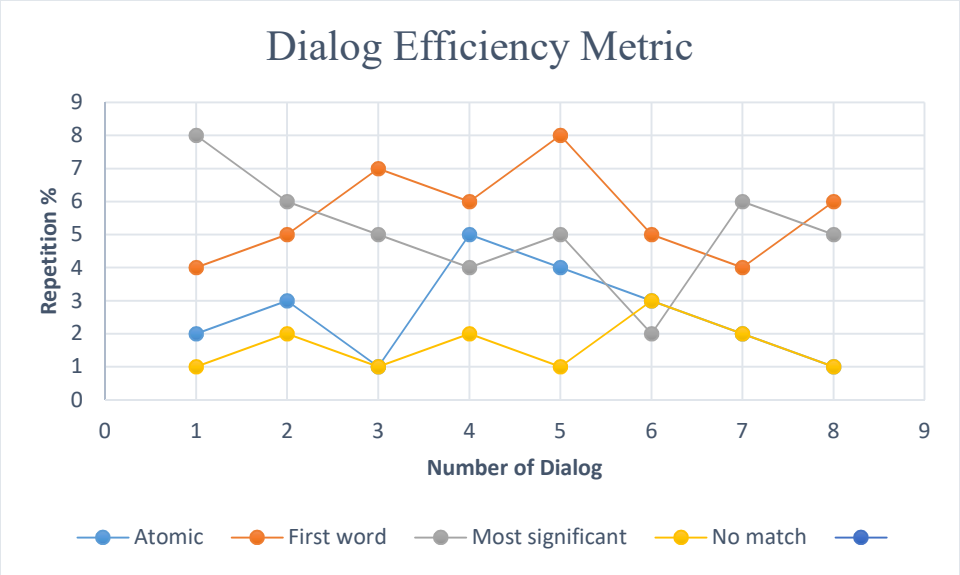


Figure 5– Dialog Efficiency Relative Frequency

6. **Conclusion**

Tamil Chatbot is one of the important systems that enables users to communicate with the system and enable to get more information about the system. In this work, a Chatbot is developed for Chennai Institute of Technology (Higher Education Institute) for providing various information about the Institute. The system is configured with Rasa Framework which is a conversational AI system. It uses the LSTM Architecture for sequencing the dialogue. The performance of the system is also relatively good compared to a regular system.

7. **References**

1. T. Kalaiyarasi, R. Parthasarathi, T.V.Geetha, (2003) Poongkuzhali-An Intelligent Tamil Chatterbot, Proceedings of Tamil Internet Conference 2003, 86 – 95.
2. Meyer von Wolff, R., Hobert, S., & Schumann, M. (2019). How may I help you? – State of the art and open research questions for chatbots at the digital workplace. Proceedings of the 52nd Hawaii International Conference on System Sciences. https://doi.org/10.24251/hicss.2019.013
3. Galitsky, B., & Goldberg, S. (2019). Explainable machine learning for chatbots. Developing Enterprise Chatbots, 53-83. https://doi.org/10.1007/978-3-030-04299-8_3.
4. Sorokin, D., & Gurevych, I. (2018). Mixing context Granularities for improved entity linking on question answering data across entity categories. Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics. https://doi.org/10.18653/v1/s18-2007.
5. Zafar, H., Napolitano, G., & Lehmann, J. (2019). Deep query ranking for question answering over knowledge bases. Machine Learning and Knowledge Discovery in Databases, 635-638. https://doi.org/10.1007/978-3-030-10997-4_41.
6. Senthilkumar, M., & Chowdhary, C. L. (2019). An AI-based chatbot using deep learning. Intelligent Systems, 231-242. https://doi.org/10.1201/9780429265020-12.
7. Rhim, J., Kwak, M., Gong, Y., & Gweon, G. (2022). Application of humanization to survey chatbots: Change in chatbot perception, interaction experience, and survey data quality. Computers in Human Behavior, 126, 107034. https://doi.org/10.1016/j.chb.2021.107034.

8. Shawar, B. A., & Atwell, E. (2007). Different measurements metrics to evaluate a chatbot system. Proceedings of the Workshop on Bridging the Gap Academic and Industrial Research in Dialog Technologies - NAACL-HLT '07. https://doi.org/10.3115/1556328.1556341.
9. https://www.interactions.com/conversational-ai/
10. Shawar, B. A., & Atwell, E. (2007). Different measurements metrics to evaluate a chatbot system. Proceedings of the Workshop on Bridging the Gap Academic and Industrial Research in Dialog Technologies - NAACL-HLT '07. https://doi.org/10.3115/1556328.1556341

Recurrent Convolutional Neural Networks for Tamil Text Classification

A.Lakshmi
Dept. of Information Technology
Chennai Institute of Technology
Email: lakshmiait2019@citchennai.net

Swasthika Venkataraman
Dept. of Information Technology
Chennai Institute of Technology
Email:swasthikavenkataramanit2019@citchennai.net

S.S. Arumugam
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ponnusamyr@citchennai.net

R. Ponnusamy
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ponnusamyr@citchennai.net

Abstract

Text classification is one of the essential function for all Information System. In the modern applications regional language plays an important role. Many Tamil Language systems are under development. In this this work an attempt is made to use the deep learning approach for Tamil text classification. In addition NLTK (Natural Language Toolkit) used as a support tool for extracting the features from the dataset. In general, the traditional classification there exist lot of noise. In this work we make an attempt to classify the Tamil text using a recurrent conventional neural network which introduce less noise compared to the traditional systems. In this we employ a max pooling that automatically judges which plays key role in text classification to capture the essential features of the text. For this design, a Wikipedia dataset collected from the Kaggle repository is taken for experimentation. Standard measurements like, accuracy, precision, recall and F1-mesuare are calculated and presented.

Keywords: Text classification, Tamil Language Systems, Natural Language Toolkit, Wikipedia, Accuracy, Precision, Recall, F1-measure.

1. Introduction

Text classification is the one of the important task in text mining and information processing. It is also challenging task while doing it for Tamil Language. Tamil text is a non-roman letter script and

therefore information processing needs additional steps to do any normal English like Language Processing. It means that it is a multi-glyph based information processing system compared to English like languages. Text Classification is the task of finding and assigning the given text into different classes. In a real time environment text analysis, organizing, sorting text data set into different classes is a time consuming task.

In the existing classification system most of the researchers use only conventional methods, like Naive Bayes, K-NN, Decision Tree, SVM (Support Vector Machine), etc. Accuracy and the semantic validity of the classified documents is the basic measure to decide the quality of the classification system. In this case of conventional methods it is basically questionable. Therefore, in the recent times it is proved that the Deep Learning algorithms played a vital role in information mining. In this work we attempt to classify the Tamil Text using the RCNN. It Recurrent Convolution Neural Network (GRU) is efficient in formulation of word ordering.

The Tamil text classification process consists of the following processes:

1. Pre-processing: This includes the removal of stop words, word roots, punctuation marks, and the removal of special characters. `

2. Characteristics extraction: It consists of a statistical methods and a linguistic methods to extract relevant characteristics from documents in order to perform a classification.

3. Processing phase: the final phase of Tamil text classification, applies the text classification system to the extracted features to classify documents into classes.

This paper made an attempt to classify the Tamil text Wikipedia documents in to 11 first level predefined categories and each categories having additional sub-categories using Tamil Wikipedia dataset collected from the Internet. The performance of the system is measured through the standard normal text classification metrics. In this paper, Section 2 reviews the literature, Section 3 describes basic News Categories of linguistic taxonomy of the Wikipedia documents. Section 4 System Architecture Design in detail. Section 5 provides an overview experimental setup in section, 6 explains the result and discussion. Section 7 concludes the paper with study of impact of the Wikipedia documents dataset.

2. Background Literature

The Tamil text classification has important tool for different mining and information retrievals system. A vast amount of work has been carried out to classify the text worldwide by considering its importance using different algorithms and modern tools. Several data sets are available on the web, leading to the development of Tamil text Data Mining. This section deliberates works related to text classification systems development.

1. Ramraj, S., et al., in 2020 [1] attempted to group the news story utilizing Convolutional neural Network strategies. It is induced that Convolutional Neural Neural with pre-prepared inserting's for Tamil language gives better classification contrasted with Support Vector Machine and Naive Bayes prepared with TFxIDF highlight vectors. The accuracy, review and F1 score for the class governmental issues is low when contrasted with other two classes.

2. Tang, X. et. al in 2019 [2], develop an multi-scale CNN based GRU network for Chinese text classification. during this except reducing the network parameters time parameters also adjusted to adapt the system. during this the medical dataset accustomed validate the effectiveness of the system.

3. Shujuan Yu, et. al., [3] attempted to classify the text using RCNN based architecture for classification of text. The results of experiments shows that the RCNN model has better precision then the other models.

4. Huaiguang Wu, Daiyi Li and Ming Cheng, in 2019[4] classified the larger Chinese text using SVM and CNN methods. The text classification is distributed with the SVM multiple classifier. The practical CSVM Algorithm gives good results while doing the Chinese text classification.

5. D. M. Harikrishna, K. Sreenivasa Rao in 2019 [5] classified stories of Indian languages using Conventional methods, like KNN and SVM. From this experiment it is giving high performance in terms of standard measurement. It is also observed that adding linguistic information boosts the performance of story classification.

6. Muhammad Zulqarnain, et. al., attempted to GRU based word embedding for text classificaiton system. with Google snippets and TREC dataset. In comparing with the traditional models like RNN, MV-RNN and LSTM this approach gives better results and error rate also good.

7. A. Kalaivania, D. Thenmozhi, in [7] 2020 attempted to classify the mixed text, that Tamil – English or Malayalam – English text using Dravidian–CodeMix-FIRE2020 Dataset. In this work AWD-LSTM model with ULMFiT framework using the FastAi library dealing with the detection and classification of sentiment.

3. Classification Categories

Wikipedia [9] is a multi-lingual encyclopedia available online in 323 languages, written and edited by community volunteer from 2001 onwards. It classifies the Tamil article in to 11 predefined Categories and the categories are shown in the following hierarchical tree. Each categories having number of sub-hierarchy. In this process of classification the given document is classified into one of the following Classification hierarchy tree shown in figure 1.

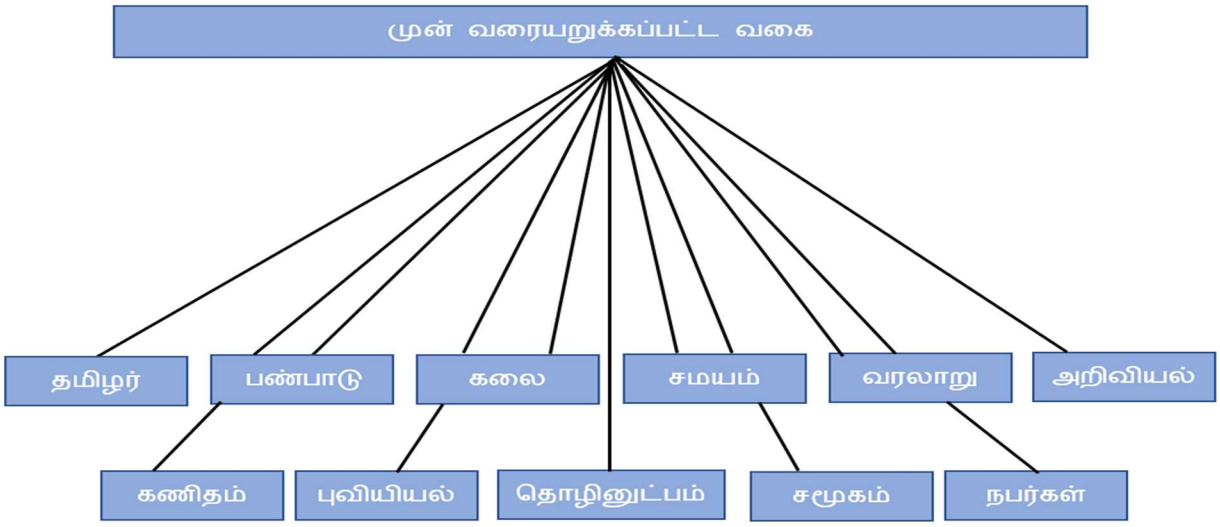


Figure 1 WikiPedia Classification Tree

4. System Design

It the system design model get the semantic value of the document. The Network architecture shown in Figure 2. The system takes the input Document D, with the word sequence w_1, w_2 . . . w_n. The output of the system gives class types and p(k|D, θ) to denote the probability of the document being class k, where θ is the parameters in the deep learning network.

Text Corpus Acquisition

A good size dataset is typically required so as to get better classification precision of any machine learning system. Tamil Wikipedia dataset is collected from the Internet. The dataset is a pre-classified into eleven primary categories, viz., Tamils, Culture, Arts, Religion, History, Science, Maths, Geography, Technology, Society and friends. Under each category there are number of sub-categories exists. The given document is pre-trained to classify the documents under these sub-categories.

Tokenization

In this case of tokenization the words are gathered from the given text corpus by matching it with the dictionary. The dictionary stores all related words or multiple words. Initially, it does a searching for the particular words or multiple words and take into account, otherwise it will be truncated. In this present setup 356,791 word entries collected from the different Internet.

Normalization

Words are converted into Tamil Unicode equivalent text and this can enable the system to grasp the identical meaning in Tamil universally in Tamil. Sometimes, Arabic alphabets, Non Standard Words may appear in the text which can be expressed as regular expression. English words also appear in the document, which are all processed and converted into Tamil text.

Stop word Elimination

There are number of functional words exists in the text, these words need to be eliminated. One such functional words collected from Internet and directly used in the system. As of now there 219 such words are identified and used in the system.

RCNN Model

The Steps are as follows:

1. Bi-directional network used instead of traditional Neural Network text representation.
2. Max-pooling layer extracts features automatically is important text classification.

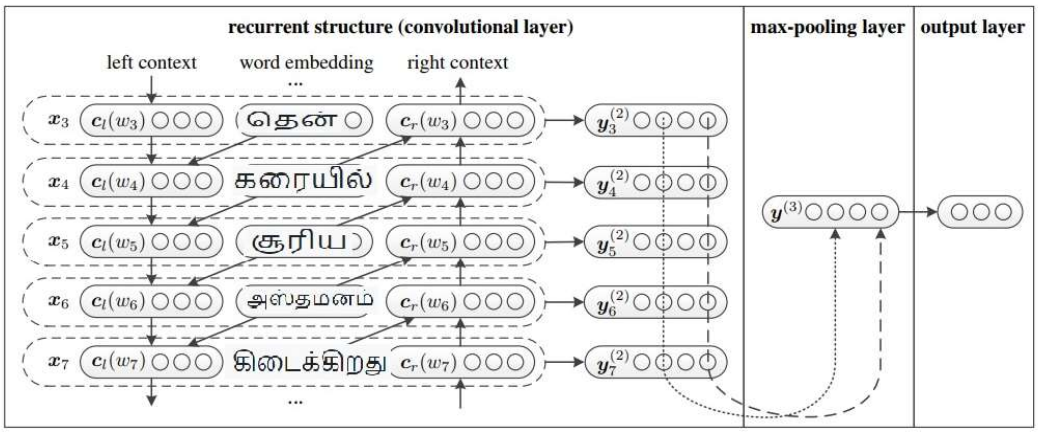


Figure 2 Architecture of Tamil Text Classification System using RCNN Model

- Input: Document D, with a word sequence w1,w2,…,wnw1,w2,…,wn.
- Output: The output of the system gives class types and p(k|D, θ) to denote the probability of the document being class k, where θ is that the parameters within the deep learning network.

Notations

• cl(wi): Context Left word wi. Dense vector R|c|.
• cr(wi): Context Right word wi. Dense vector R|c|.
• e(wi): Embedding Word wi pre-trained with the Skip-gram model. Dense vector R|e|.
• cl(w1): Context Left of the primary word w1 in given documents.
• cr(wn): Context Right of the last word wn in given documents.
• W(l): Matrix that transforms the hidden layer (context) into the subsequent hidden layer.
• W(sl): Matrix that accustomed combine the semantic of the present word with the subsequent word's left context.
• f: activation function with non-linear.

Equations

$$cl(wi)=f(W^{(l)}cl(wi-1)+W^{(sl)}e(wi-1)) \quad \text{----------(1)}$$

$$cr(w_i)=f(W^{(r)}cr(w_{i+1})+ W^{(sr)}e(w_{i+1}))$$ ----------(2)

- **(1),(2)** show how to compute context vectors cl(wi),cr(wi).Note that the equations are slightly different from vanilla RNN:

$$h_t=f(W_{xh}x_t+W_{hh}h_t-1)$$

$$x_i=[cl(w_i);e(w_i);cr(w_i)]$$ ---------- (3)

- **(3)** shows how to represent a word w_i

$$y_i^{(2)} = \tanh(W^{(2)}x_i+b^{(2)})$$ ---------- (4)

- **(4)** shows how to compute latent semantic vector $\mathbf{y_i^{(2)}}$ in which semantic factor will be analyzed to the most useful factor for representing the text

Text Representation Learning

Use max-pooling layer converts texts with various length into a fixed-length vector.

Equations

$$Y^{(3)}=max^n{}_{i=1}\, y_i^{(2)}$$ ---------- (5)

- (5) Exhibit the method to perform max-pooling. The k-th element of $y^{(3)}$ is the maximum in the k-th elements of $y_i^{(2)}$. The max pooling layer attempts to find the most important latent semantic factors in the document.

$$Y^{(4)} = W^{(4)}y^{(3)} +b^{(4)}$$ ----------(6)

- **(6)** is output layer similar to traditional NN

$$Pi = \frac{\exp(yi(4))}{\sum_{k=1}^{n}\exp(yk(4))}$$ ---------- (7)

- **(7)** applies softmax function to convert the output numbers into probabilities

Training

Size Variables

- **|V|**: Vocabularyy size
- H: Hidden layer size
- **O**: Number of classes in this case it is 11.

Paramaters

- **E:** Word embeddings $\mathbf{R}^{|e|x|V|}$

- $\mathbf{b^{(2)}}$:Bias vector $\mathbf{R^H}$
- $\mathbf{b^{(4)}}$: Bias vector $\mathbf{R^O}$
- $\mathbf{cl^{(wl)}, cr^{(wn)}}$: Initial contexts $\mathbf{R^{|c|}}$
- $\mathbf{W^{(2)}}$: Transformation matrix $\mathbf{R^{H\times(|e|+2|c|)}}$
- $\mathbf{W^{(4)}}$:Transformation matrix $\mathbf{R^{O\times H}}$.
- $\mathbf{W^{(l)}, W^{(r)}}$: $\mathbf{R^{|c|x|c|}}$
- $\mathbf{W^{(sl)}, W^{(sr)}}$: $\mathbf{R^{|e|x|c|}}$

Objective

$$\text{Arg max} \sum_{D\epsilon D} \log p(class\ D|D,\theta)$$

It is our objective to log-likelihood maximize the with respect to θ, where D is the training document set and $class_D$ is the correct class of document D

Optimizer

$$\boldsymbol{\theta \leftarrow \theta + \alpha \frac{\partial\ log\ p\ (classD|D,\theta)}{\partial,\theta}}$$

Use stochastic gradient descendent to optimize objective

5. Experimental

The dataset used for classification is received from the Kaggle free open source repository. It is a Tamil Wikipedia Document Dataset of eleven different categories. In this dataset includes four different attributes, such as Document title and Document. These documents are collected and manually classified before feeding into the system. In this dataset the given dataset is divided into two groups. One group for training and another one group for test data.

5.1 Experiment Setup

A python program is developed with the Pytorch, nltk, Sklearn, Pandas, NumPy packages to model the RCNN model to predict the predefined text categories. This prototype works well for state-of-the-art data in the database. In this program, it will be able to expect only the RCNN model alone. The system is configured to run in a standalone PC Environment.

5. Results and Discussion

The performance of the Tamil text classification arrangement can be appraised by using four different standard metrics that is Precision, Recall, Accuracy, and F1 measure. Precision dealings the factualness of the classifier system. A higher precision means less false positives, while a lower precision means false positives.

$$\text{Precison} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of extracted text}}$$

Recall measures the completeness, or sensitivity, of a classifier. Higher recall means less false negatives, while lower recall means more false negatives.

$$\text{Precison} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of annotated text}}$$

The correct classification instance measured by,

$$\text{Accuracy} = \frac{Number\ of\ correctly\ classified\ instances}{\text{Total Number of instances}}$$

A harmonic F1-measure is finding mean of recall and precision

$$\text{F1} - \text{Mesure} = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 1 Performance of Results Evaluation of RCNN System Testing Process

Label	Precision	Recall	F1-Score	Accuracy
1	92	90	90.99	91
2	90	91	90.5	92
3	91	92	91.5	93
4	92	93	92.5	91
5	93	91	91.99	90
6	91	90	90.5	91
7	89	91	89.99	92
8	92	92	92	91
9	90	93	91.48	92
10	91	91	91	93
11	92	93	94	95
Average	91.18	91.54	91.49	91.90

6. Conclusion

Tamil Text Classification is the one of the essential task in the Text Mining and Natural Language Processing. Developing and acquiring more accurate results for Text Classification is the challenging task. This paper proposes a novel RCNN model for Tamil Text Classification into a fifteen pre-defined categories. In this method the classification accuracy is 91 percent and above which is comparatively good as compared to the traditional classifiers.

7. References

1. Ramraj, S., Arthi, R., Murugan, S., & Julie, M. (2020). Topic categorization of Tamil news articles using PreTrained Word2Vec embeddings with Convolutional neural network. 2020 International

Conference on Computational Intelligence for Smart Power System and Sustainable Energy (CISPSSE). https://doi.org/10.1109/cispsse49931.2020.9212248.

2. Tang, X., Chen, Y., Dai, Y., Xu, J., & Peng, D. (2019). A multi-scale Convolutional attention based GRU network for text classification. 2019 Chinese Automation Congress (CAC). https://doi.org/10.1109/cac48633.2019.8996433.
3. Siwei Lai, Liheng Xu, Kang Liu, Jun Zhao, (2015), Recurrent Convolutional Neural Networks for Text Classification, Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, https://www.deeplearningitalia.com/wp-content/uploads/2018/03/Recurrent-Convolutional-Neural-Networks-for-Text-Classification.pdf
4. Wu, H., Cheng, M., & Li, D. (2019). Chinese text classification based on character-level CNN and SVM. International Journal of Intelligent Information and Database Systems, 12(3), 212. https://doi.org/10.1504/ijiids.2019.10024507.
5. Harikrishna, D. M., & Rao, K. S. (2020). Children's story classification in Indian languages using linguistic and keyword-based features. ACM Transactions on Asian and Low-Resource Language Information Processing, 19(2), 1-22. https://doi.org/10.1145/3342356.
6. Zulqarnain, M., Ghazali, R., Ghouse, M. G., & Mushtaq, M. F. (2019). Efficient processing of GRU based on word embedding for text classification. JOIV : International Journal on Informatics Visualization, 3(4). https://doi.org/10.30630/joiv.3.4.289.
7. Kalaivania, D. Thenmozhi, (2020), @Dravidian-CodeMix-FIRE2020: Sentiment Code-Mixed Text Classification in Tamil and Malayalam using ULMFiT, FIRE 2020: Forum for Information Retrieval Evaluation, December 16–20, 2020, Hyderabad, India.
8. www.kaggle.com

Tamil News Text Classification using Gated Recurrent Unit

S.Sruthi
Dept. of Information Technology
Chennai Institute of Technology
Email: sruthisit2019@citchennai.net

Sai Keerthana A
Dept. of Information Technology
Chennai Institute of Technology
Email: Attotasaikeerthanait2019@citchennai.net

S. Pavithra
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail:pavithras@citchennai.net

R. Ponnusamy
Center for Artificial Intelligence & Research
Chennai Institute of Technology
E-mail: ponnusamyr@citchennai.net

Abstract

Tamil Text classification is the most important task required in different tools and systems design directly or indirectly. There are several methods available for Tamil text classification. Most of the existing classification methods suffer in terms of design, speed, accuracy and etc. In order to overcome this problem in this research an attempt is made to design a classification system with Gated Recurrent Unit, which is faster, adaptive, etc. Also, the GRU's enactment on certain tasks of text classification, polyphonic music modeling, speech signal processing and natural language processing is proved to be the best system. It comprises of the reset gate and the update gate instead of the input, output and forget gate. In this process of classification, at first the nouns, verbs are extracted and the term-frequency, inverse document frequency are taken from the training and testing documents by eliminating the stop words. This input is given to the Gated Recurrent Unit machine to generate classes. At first the system uses the Tamil daily newspaper Dailythanthi dataset collected from the newspaper data for both training and testing. The standard measures like, precision, recall and F1-mesuare are computed presented.

Keywords: Tamil text classification, speed, accuracy, Gated Recurrent Unit, Precision, Recall, F1 measure.

1. Introduction

Text classification is the one of the important task in text mining and information processing. It is also challenging task while doing it for Tamil Language. Tamil text is a non-roman letter script and therefore information processing needs additional steps to do any normal English like Language Processing. It means that it is a multi-glyph based information processing system compared to English like languages. Text Classification is the task of finding and assigning the given text into different classes. In a real time environment text analysis, organizing, sorting text data set into different classes is a time consuming task.

In the existing classification system most of the researchers use only conventional methods, like Naive Bayes, K-NN, Decision Tree, SVM (Support Vector Machine), etc. Accuracy and the semantic validity of the classified documents is the basic measure to decide the quality of the classification system. In this case of conventional methods it is basically questionable. Therefore, in the recent times it is proved that the Deep Learning algorithms played a vital role in information mining. In this work we attempt to classify the Tamil Text using the GRU Machine. It Gated Recurrent Unit (GRU) is a gating instrument in RNN gifted with learning order dependency in arrangement forecast difficulties.

The Tamil text classification process consists of the following processes:

1. Pre-processing: This includes the removal of stop words, word roots, punctuation marks, and the removal of special characters. `

2. Characteristics extraction: It consists of a statistical methods and a linguistic methods to extract relevant characteristics from documents in order to perform a classification.

3. Processing phase: the final phase of Tamil text classification, applies the text classification system to the extracted features to classify documents into classes.

This paper made an attempt to classify the Tamil text documents in to 7 first level predefined categories and each categories having additional sub-categories using Tamil daily newspaper Dinamalar dataset collected from the Internet. The performance of the system is measured through the standard normal text classification metrics. In this paper, Section 2 reviews the literature, Section 3 describes basic News Categories of linguistic taxonomy of the Dinamalar News Paper. Section 4 System Architecture Design in detail. Section 5 provides an overview experimental setup in section, 6 explains the result and discussion. Section 7 concludes the paper with study of impact of the News Paper.

2. Background Literature

The Tamil text classification has important tool for different mining and information retrievals system. A vast amount of work has been carried out to classify the text worldwide by considering its importance using different algorithms and modern tools. Several data sets are available on the web, leading to the development of Tamil text Data Mining. This section deliberates works related to text classification systems development.

1. Ramraj, S., et al., in 2020 [1] attempt to group the news story utilizing Convolutional neural organization strategies. It is induced that Convolutional neural organization with pre-prepared inserting's for Tamil language gives better classification contrasted with Support Vector Machine and Naive Bayes prepared with TFxIDF highlight vectors. The accuracy, review and F1 score for the class governmental issues is low when contrasted with other two classes.

2. Tang, X. et. al in 2019 [2], develop an multi-scale CNN based GRU network for Chinese text classification. during this except reducing the network parameters time parameters also adjusted to adapt the system. during this the medical dataset accustomed validate the effectiveness of the system.

3. Shujuan Yu, et. al., [3] attempted to classify the text using LSTM, GRU and CNN based architecture for classification of text. The results of experiments shows that the ABLGCNN model has faster convergence speed and better precision then the opposite models.

4. Huaiguang Wu, Daiyi Li and Ming Cheng, in 2019[4] classified the larger Chinese text using SVM and CNN methods. The text classification is distributed with the SVM multiple classifier. The experimental results show that the CSVM algorithm is simpler than other traditional Chinese text classification algorithm.

5. D. M. Harikrishna, K. Sreenivasa Rao in 2019 [5] classified stories of Indian languages using Conventional methods, like KNN and SVM. It's observed that Performance of the classifier is evaluated using 10-fold cross-validation and effectiveness of classifier is measured using precision, recall, and F-measure. From the classification results, it's observed that adding linguistic information boosts the performance of story classification.

6. Muhammad Zulqarnain, et. al., attempted to GRU based word embedding for text classificaiton system. with Google snippets and TREC dataset. In comparing with the traditional models like RNN, MV-RNN and LSTM this approach gives better results and error rate also good.

7. A. Kalaivania, D. Thenmozhi, in [7] 2020 attempted to classify the mixed text, that Tamil – English or Malayalam – English text using Dravidian–CodeMix-FIRE2020 Dataset. In this work AWD-LSTM model with ULMFiT framework using the FastAi library dealing with the detection and classification of sentiment.

3. Classification Categories

Dinamalar [9] is a daily evening Tamil newspaper. It is being published in various parts of Tamilnadu from 1951 onwards. It classifies the news article in to 7 Categories and the categories are shown in the following hierarchical tree. Each categories having number of sub-hierarchy. In this process of classification the given document is classified into one of the following Classification hierarchy tree shown in figure 1.

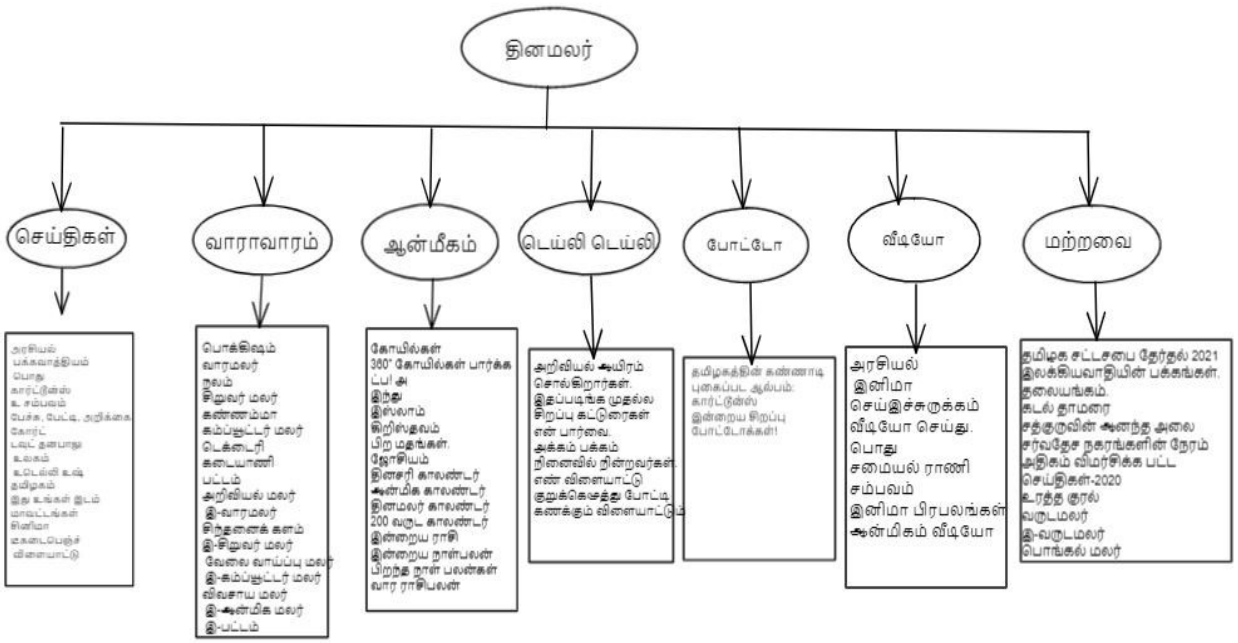


Figure 1 Dinamalar Classification Tree

4. System Design

In planned model first the Input of each document is estimated to identify the classes. The input variable predicted through GRU model is the critical factor of Dinamalar Text data set.

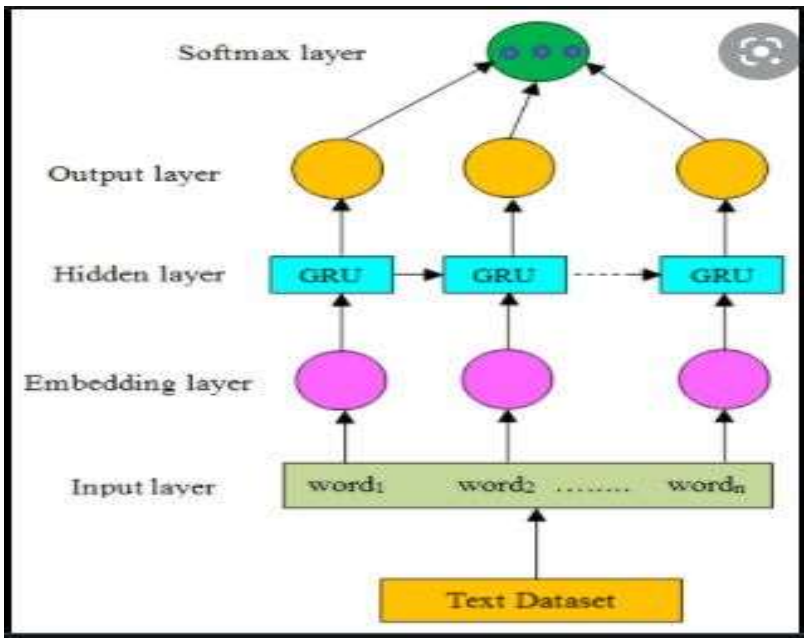


Figure 2 Architecture of Tamil Text Classification System using GRU Model

Text Corpus Acquisition

A good size dataset is typically required so as to get better classification precision of any machine learning system. Dinamalar dataset is collected from the Internet. The dataset is a pre-classified into seven primary categories, viz., News, Daily, Weekly, Spiritual, Photo, video and others. Under each category there are number of sub-categories exists. The given document is pre-trained to classify the documents under these sub-categories.

Tokenization

In this case of tokenization the words are gathered from the given text corpus by matching it with the dictionary. The dictionary stores all related words or multiple words. Initially, it does a searching for the particular words or multiple words and take into account, otherwise it will be truncated. In this present setup 356,791 word entries collected from the different Internet.

Normalization

Words are converted into Tamil Unicode equivalent text and this can enable the system to grasp the identical meaning in Tamil universally in Tamil. Sometimes, Arabic alphabets, Non Standard Words may appear in the text which can be expressed as regular expression. English words also appear in the document, which are all processed and converted into Tamil text.

Stop word Elimination

There are number of functional words exists in the text, these words need to be eliminated. One such functional words collected from Internet and directly used in the system. As of now there 219 such words are identified and used in the system.

GRU Model

It is kind of neural network which is easier to recollect the past memory known as Gated Recurrent Unit Memory (GRU). In GRU is like LSTM, which has two gates, reset gate and update gate and it uses hidden state to transfer the information one state to another state.

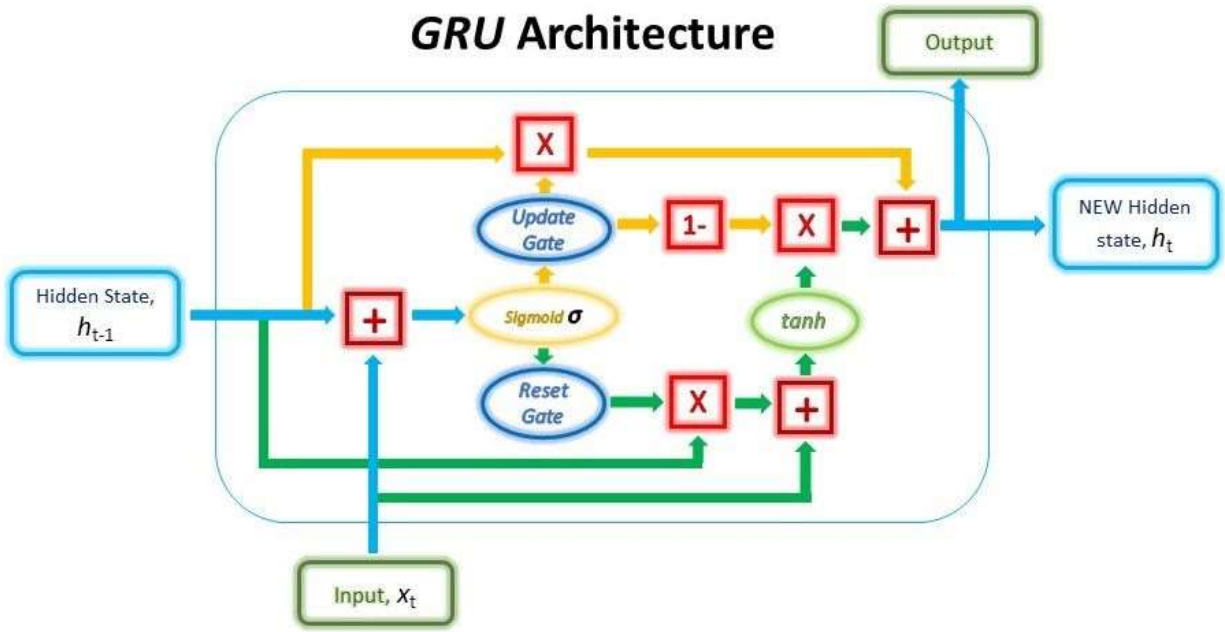


Figure 3 GRU gates

1. **Reset gate** – it gets the previous hidden time stamp and multiply with the present input and weight and after that it will pass it to the sigmoid function. The sigmoid function transform values between 0 to 1.

$$gate_{reset} = \sigma(W_{input_{reset}} \cdot x_t + W_{hidden_{reset}} \cdot h_{t-1})$$

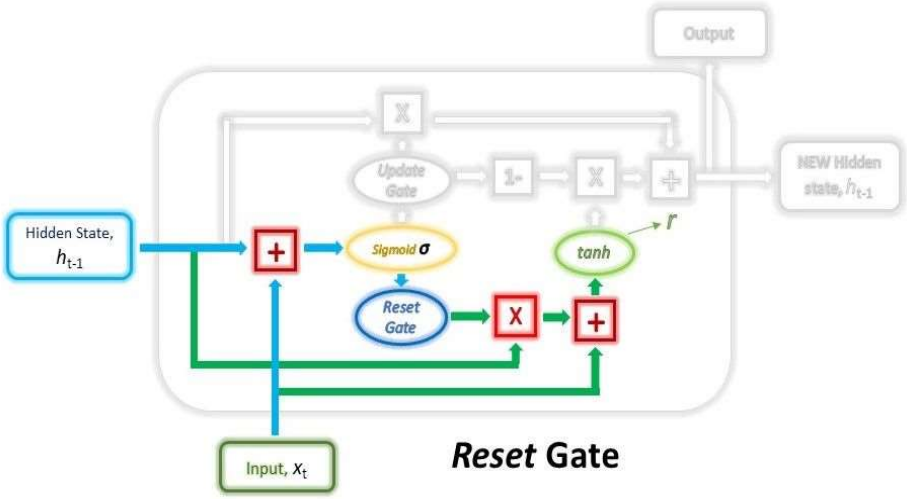
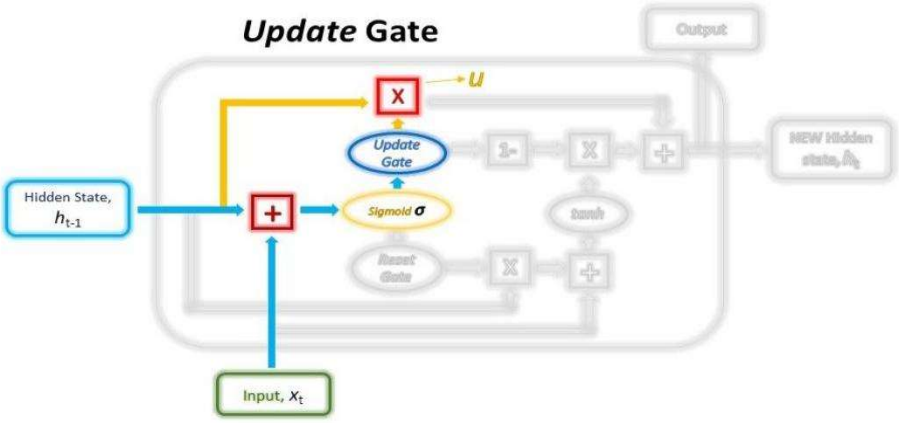


Figure 4 Reset Gate

In the hidden-state the trainable weight used for multiplying with element wise reset vector. It will confine which values of previous time stamp can be retrained or eliminated from the hidden state. At last a non-linear activation tanh function are applied and the result is obtained.

2. Update gate: As like Reset gate, update gate using the same formulae, otherwise the weight given to multiply with the will vary and it is unique for different hidden states. This enable the gates to serve as unique one.



$$z_t = \sigma_g\,(W^{(z)}x_t + U^{(z)}h_{t-1} + b^{(z)})$$

Figure 5 Update gate

3. **Candidate state**

$$\hat{h}t = \tanh\,(W(\hat{h})xt) + U(\hat{h})(rt * ht\text{-}1)$$

4. **Final Output**

$$ht = zt * ht\text{-}1 + (1 - zt) * \hat{h}t\text{-}1$$

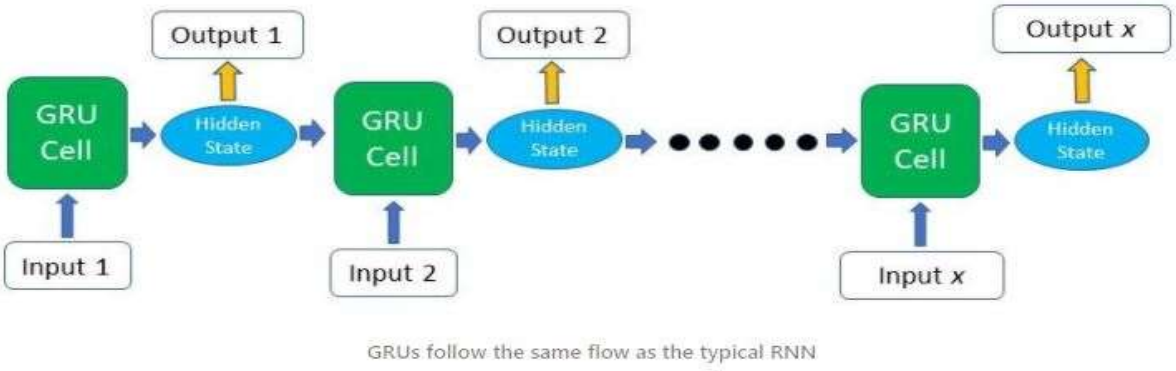


Figure 6 GRU Chain Organization Model

5. **Experimental**

The dataset used for classification is received from the Kaggle free open source repository. It is a Dinalar News Dataset News Articles of fifteen different categories. In this dataset includes four different attributes, such as News Id, News Category, News Title and News. In this dataset the

given dataset is divided into two groups. One group for training and another one group for test data. In case of training all attributes data items are taken and in case of testing the News Title and News Article alone take for experimentation.

5.1 Experiment Setup

A python program is developed with the Keras, nltk, Scikit-learn, Pandas, NumPy, Tensorflow packages to model the GRU model to predict the predefined text categories. This prototype works well for state-of-the-art data in the database. In this program, it will be able to expect only the GRU model alone. The system is configured to run in a standalone PC Environment.

5. Results and Discussion

The performance of the Tamil text classification system can be evaluated by using four different standard metrics that is Accuracy, Precision, Recall and F1 measure. Precision measures the factualness of the classifier system. A higher precision means less false positives, while a lower precision means false positives.

$$\text{Precison} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of extracted text}}$$

Recall measures the completeness, or sensitivity, of a classifier. Higher recall means less false negatives, while lower recall means more false negatives.

$$\text{Precison} = \frac{Number\ of\ correct\ extracted\ text}{\text{Total number of annotated text}}$$

Accuracy measures the overall degree of which instances have been correctly classified,

$$\text{Accuracy} = \frac{Number\ of\ correctly\ classified\ instances}{\text{Total Number of instances}}$$

A weighted harmonic measure mean of precision and recall is F1-measure, is the rate of a system with one unique rating.

$$\text{F1} - \text{Mesure} = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 1 Performance of Results Evaluation of GRU System Testing Process

Label	Precision	Recall	F1-Score	Accuracy
1	97	98	99	98
2	97	98	99	98
3	97	98	99	98
4	98	98	99	98
5	96	97	97	97
6	97	98	99	98
7	96	96	93	96
Average	96.85	97.57	97.85	97.57

6. Conclusion

Tamil Text Classification is the one of the essential task in the Text Mining and Natural Language Processing. Developing and acquiring more accurate results for Text Classification is the challenging task. This paper proposes a novel GRU model for Tamil Text Classification into a fifteen pre-defined categories. In this method the classification accuracy is 97 percent and above which is comparatively good as compared to the traditional classifiers.

7. References

1. Ramraj, S., Arthi, R., Murugan, S., & Julie, M. (2020). Topic categorization of Tamil news articles using PreTrained Word2Vec embeddings with Convolutional neural network. 2020 International Conference on Computational Intelligence for Smart Power System and Sustainable Energy (CISPSSE). https://doi.org/10.1109/cispsse49931.2020.9212248.
2. Tang, X., Chen, Y., Dai, Y., Xu, J., & Peng, D. (2019). A multi-scale Convolutional attention based GRU network for text classification. 2019 Chinese Automation Congress (CAC). https://doi.org/10.1109/cac48633.2019.8996433.
3. Yu, S., Liu, D., Zhu, W., Zhang, Y., & Zhao, S. (2020). Attention-based LSTM, GRU and CNN for short text classification. Journal of Intelligent & Fuzzy Systems, 39(1), 333-340. https://doi.org/10.3233/jifs-191171.
4. Wu, H., Cheng, M., & Li, D. (2019). Chinese text classification based on character-level CNN and SVM. International Journal of Intelligent Information and Database Systems, 12(3), 212. https://doi.org/10.1504/ijiids.2019.10024507.
5. Harikrishna, D. M., & Rao, K. S. (2020). Children's story classification in Indian languages using linguistic and keyword-based features. ACM Transactions on Asian and Low-Resource Language Information Processing, 19(2), 1-22. https://doi.org/10.1145/3342356.
6. Zulqarnain, M., Ghazali, R., Ghouse, M. G., & Mushtaq, M. F. (2019). Efficient processing of GRU based on word embedding for text classification. JOIV : International Journal on Informatics Visualization, 3(4). https://doi.org/10.30630/joiv.3.4.289.
7. Kalaivania, D. Thenmozhi, (2020), @Dravidian-CodeMix-FIRE2020: Sentiment Code-Mixed Text Classification in Tamil and Malayalam using ULMFiT, FIRE 2020: Forum for Information Retrieval Evaluation, December 16–20, 2020, Hyderabad, India.
8. www.kaggle.com
9. www.dinamalar.com

SENTIMENT ANALYSIS AND OPINION MINING FOR TAMIL TEXT

Sanjjushri Varshini R

2 nd year,B.Tech Artifical Intelligence and Data Science

Chennai Institue of Technology

Email: sanjjushrivarshini@gmail.com

Priyadharshini R

2 nd year,B.Tech Artifical Intelligence and Data Science

Chennai Institute of Technology

Email: priya35471@gmail.com

A K Supriya

2 nd year, B.Tech Artificial Intelligence and Data Science

Chennai Institute of Technology

Email: supriyakumarvel2003@gmail.com

Marimuthu Madhana Laxmi

2 nd year,B.Tech Artifical Intelligence and Data Science

Chennai Institute of Technology

Email: mahalaxmi2807@gmail.com

Dr.J.Venkatesh

Professor, Centre for System Design, Chennai Institute of Technology

Email: venkateshj@citchennai.com

Abstract:

Sentimental Analysis is a type of the processing of Natural Language Processing (NLP) to track the public mood to a certain law, policy, or marketing, etc, which are posted on social media. The overall growing importance of sentiment analysis is connected with the development of social media such as reviews, Twitter, blogs, micro-blogs, forum discussions, e-commerce websites, and other social networks. In day-to-day life, it is difficult to analyze all the reviews in another language (Tamil) manually, so we came up with the idea of sentiment analysis for Tamil text. It is not only enough to analyze the English text, we need to consider other language reviews to come up with the exact result of the particular product. Most of the people just ignore the Tamil language reviews, we focused on it to give importance to Tamil language opinions. The main aim presents a survey of sentiment analysis (SA) and opinion mining (OM) approaches, various techniques used that are related to this field for Tamil text**.**

Keywords: Sentiment Analysis, Natural Language Processing, Tamil Language, Opinion Mining. to

Introduction:

Sentiment analysis and opinion mining are machine learning subfields. They are crucial in the current situation because there are so many user-generated texts available. Natural language is extremely unstructured and hence it is a difficult problem to solve. It is exhausting to work with a machine as the interpretation of the meaning of a particular sentence by a

The machine is tiresome. . However, the use of sentimental analysis is increasing day by day. Every day, the amount of analysis grows. Machines have to be built in terms of their ability to interpret and comprehend, it is dependable and efficient. Emotions and sensations in humans Analysis of public opinion and opinion mining is a method of implementing the same.

Because of the expansion and advancement of informatics, many e-Commerce websites exist where individuals may discuss their thoughts on items and services. Consumer feedback is a treasure trove of information in the present age. For both businesses and people, there are thousands of products to choose from On the internet that several merchants have presented. For example,

flipkart.com registers a total of more than 36 million products, Shoper.com records more than 5 million products from over 3,000 dealers. The user‟s hunger is for and dependence upon online advice and recommendations the data reveals is merely one reason behind the emergence of interest in new systems that deal directly with opinions as a first-class object. The user's desire for and reliance on online advice and recommendation is just one reason for the growing interest in novel systems that deal directly with opinions as a first-class object.

Reasons WHY we need this project:

Sentiment analysis is becoming a crucial tool for monitoring and understanding client sentiment as they share their opinions and feelings more openly than ever before. Brands can learn what makes customers happy or frustrated by automatically evaluating customer feedback, such as comments in survey replies and social media dialogues. This allows them to customise products and services to match their customers' demands.

For example, employing sentiment analysis to examine 4,000+ customer satisfaction surveys about your product could help you figure out if customers like your pricing and customer service.

Perhaps you'd like to track brand sentiment on social media in real-time and over time so you can see angry customers right away and respond quickly.

Hence it is much more important to analyse the sentiment of text in the Tamil language also.

The multinomial Naive Bayes classification algorithm is commonly used as a starting point for sentiment analysis. The core idea behind the Naive Bayes technique is to use the joint probabilities of words and classes to find the probability of classes given to texts. So we have used the Naive Bayes model.

Dataset Description:

A glimpse of data:

reviews	book title	sentiment
இந்த புத்தகம் உண்மையில் அதிக கவலை பிரச்சனை உள்ளவர்களுக்கு அவசியம். இந்த புத்தகம் நிலைமையை எவ்வாறு நிர்வகிப்பது மற்றும் நாம் எப்படி முடிவுகளை எடுக்க வேண்டும் என்பதை வெளிப்படுத்துகிறது. நீங்கள் அதற்கு செல்லலாம். பணத்திற்கு மதிப்பு.	Emotional Intelligence (Tamil)	positive
தமிழில் நல்ல புத்தகம். அடிப்படை இதை கற்றுக்கொள்ள முடியும். ஆனால் இந்த புத்தகங்களில் விமர்சன நுண்ணறிவு இல்லை. புதியவர்களாக. முன்முயற்சி எடுத்த ஆசிரியருக்கு நன்றி.	Emotional Intelligence (Tamil)	negative

We collected our data in a short duration of time from media like Amazon, Flipkart, Goodreads, etc. We focussed only on Tamil book reviews in the Tamil language from different users and different genres of books. Our dataset consists of rows around 400 which have mixed feelings from the user i.e., positive or negative.

Sentimental Classification:

Much study has been done on user sentiment analysis, which mostly judges the polarities of user evaluations. Sentiment analysis is frequently used in this research at one of three levels: document level, phrase level, or attribute level. According to the literature review, there are two types of sentiment analysis techniques: machine learning and semantic orientation. Furthermore, natural language processing techniques (NLP) are employed in this field, particularly in the identification of document sentiment. The sentiment analysis machine learning strategy is largely related to supervised classification in general and text classification techniques in particular. As a result, it's known as "supervised learning." A machine learning-based categorization requires two sets of documents: a training set and a test set. An automatic classifier learns the differentiating properties of documents using a training set, while a test set is used to assess the automatic classifier's performance. To categorise the reviews, a variety of machine learning approaches were used.

Machine learning techniques like Naive Bayes (NB) have achieved great success in text categorization. The other most well-known machine learning methods in the natural language processing area are K-Nearest neighbourhood, ID3, C5, centroid classifier, winnow classifier, and the N-gram model. Naive Bayes is a simple but effective classification algorithm. The Naive Bayes algorithm is a widely used algorithm for document classification. The basic idea is to estimate the probabilities of categories given in a test document by using the joint probabilities of words and categories. The naive part of such a model is the assumption of word independence. The simplicity of this assumption makes the computation of the Naive Bayes classifier more Efficient.

Sentiment classification is accomplished by building a text classifier from association rules that link the terms of a document to its categories, and then modelling the text documents as a collection of transactions, with each transaction representing a text document and the items in the transaction representing the terms selected from the document and the categories to which the document is assigned. The algorithm then looks for connections between the words in documents and the labels that have been assigned to them. Each category is treated as a distinct text collection to which association rule mining is applied. The classifier is formed by combining the rules generated by each of the categories independently. The training set is then utilised to evaluate the classification quality, and the number of rules covered and attributive probability are used to categorise the test text documents. Using generalised expectation criteria, the labelled features are used to constrain the model's predictions on unlabeled cases. The un-annotated text is then supplied into the self-learned features extractor, and the documents labelled with high confidence are fed into the initially-trained classifier using generalised expectation to acquire domain-dependent features automatically. Following that, the self-learned features are used to train a new classifier, which is then applied to the test set to get the final results.

A few recent studies in this field explained the use of neural networks in sentiment classification. Zhu Jian (2010) proposed an individual model based on Artificial neural networks to divide the movie review corpus into positive, negative and fuzzy tones which are based on the advanced recursive least squares backpropagation training algorithm. Long-Sheng Chen (2011) proposed a neural network-based approach, which combines the advantages of machine learning techniques and information retrieval techniques.

Semantic Orientation:

Since it does not require prior training to mine the data, the Semantic orientation approach to Sentiment analysis is regarded as "unsupervised learning." Instead, it assesses a word's proclivity for positive and negative connotations. Many studies on unsupervised sentiment categorization make use of the lexical resources that are accessible.

Role of negation:

Negation is a prevalent linguistic construction that influences polarity and, as a result, must be taken into account in sentiment analysis. Not only do typical negation words (not, neither, nor) express negation, but so do other lexical components. Many more words, such as valence shifters, connectives, and modals, have been discovered to invert the polarity of an articulated opinion, according to field research. “I find

the functioning of the new mobile less practical," as an example of valence shifter; "Perhaps it is a terrific phone, but I fail to see why," as an example of connectives' effect. "In principle, the phone should have worked even underwater," is an example of a modal statement. Negation is a challenging but vital part of sentiment analysis, as these examples demonstrate. The most well-known work in sentiment analysis is an examination of the influence of various scope models for negation. Using static delimiters, dynamic delimiters, and heuristic rules centred on polar expressions, a scope identification approach to handle negation was developed. Static delimiters are clear words that mark the beginning of another phrase, such as because or unless. Dynamic delimiters, on the other hand, are rules that rely on contextual information such as the part-of-speech tag. These delimiters appropriately account for a variety of complex phrase patterns, ensuring that only the clause containing the negative is taken into account. The heuristic rules focus on cases in which polar

expressions in specific syntactic configurations are directly preceded by negation words which result in the polar expression becoming a delimiter itself.

Sentiment analysis has been a trendy issue in data mining, with extracting opinion features being a vital step, thanks to the growing amount of opinions and reviews on the internet. Sentiment analysis has been too coarse at both the document and sentence levels to discern precisely what users like and dislike. To overcome this issue, sentiment analysis at the attribute level aims to extract opinions on specific attributes of products from reviews.

Online advertising, hotspot detection in forums, and some applications of sentiment analysis. Online advertising has evolved into one of the most important revenue streams in today's Internet economy. Sentiment analysis has recently been used in dissatisfaction-based internet advertising and Blogger-Centric Contextual Advertising, which refers to the placement of personal ads on any blog page based on the interests of the bloggers. When confronted with massive amounts of online material from a variety of online forums, information searchers sometimes find it challenging to extract reliable and valuable information. This has prompted a study into identifying online forum hotspots where relevant information is swiftly disseminated to individuals seeking it. Nan Li (2010) developed a sentiment analysis approach to provide a complete and timely description of the interacting structural natural groupings of distinct forums, allowing for dynamically efficient hotspot forum finding. Companies must acquire and analyse information about their competitors' products and plans to identify potential hazards. Sentiment analysis plays an important role in competitive intelligence by extracting and visualising comparative relations between products from customer reviews, taking into account interdependencies among relations, to assist businesses in identifying potential risks and developing new products and marketing strategies. Opinion summarization analysis articles' viewpoints by indicating sentiment polarities, degrees, and associated events. With an opinion summary, a customer may quickly learn how other customers feel about a product, and the manufacturer can learn why various individuals enjoy it or what they dislike about it. Opinion extraction algorithms at the word, phrase, and paragraph level are offered. The topic of relevant sentence selection is discussed, followed by a summary of topical and opinionated material. Opinion summarizations are visualized by representative sentences. Finally, an

opinionated curve showing supportive and non-supportive degrees along the timeline is illustrated by an opinion tracking system. Other applications of sentimental analysis include online message sentiment, filtering-mail sentiment classification, web blog author‟s attitude analysis etc.

Naïve Bayes :

One of the most effective algorithms for categorising documents is Naive Bayes24. It has been widely employed in the field of information recovery, and it has recently been applied in machine learning research25. The Bernoulli model and the multinomial model have gotten a lot of attention in recent years26. The integer feature is represented by the multinomial template to represent the document, whereas the Bernoulli model vector of binary features is obtained from the document27.

To apply a basic assumption to the Bayes' theorem, namely, feature independence. So now we've separated the evidence into its component elements.

If any two occurrences A and B are independent of one another, then

P(A,B) = P(A)P(B)

Hence, we reach the result:

$$P(y|x_1,...,x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

which can be expressed as:

$$P(y|x_1,...,x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1)P(x_2)...P(x_n)}$$

Now, as the denominator remains constant for a given input, we can remove that term:

$$P(y|x_1,...,x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

Multinomial Naïve bayes multinomial classification:

Multinomial naive Bayes is a specific variant of naive Bayes that is designed to handle text documents and uses word counts as the underlying mechanism of probability calculation. It's a basic yet straightforward methodology for dealing with classification tasks that don't need sentiment analysis (complex expressions of emotions such as sarcasm). For example, you have a collection of classes that represent school topics based on an arbitrary document inside your set of classes (mathematics, art, music, etc..).You are given a document with words that are strongly connected to art since they utilised a lot of keywords, but there may also be some mathematics, history, and other subjects, but after going through all of the classes and calculating their probabilities, it comes out that it is most likely art.

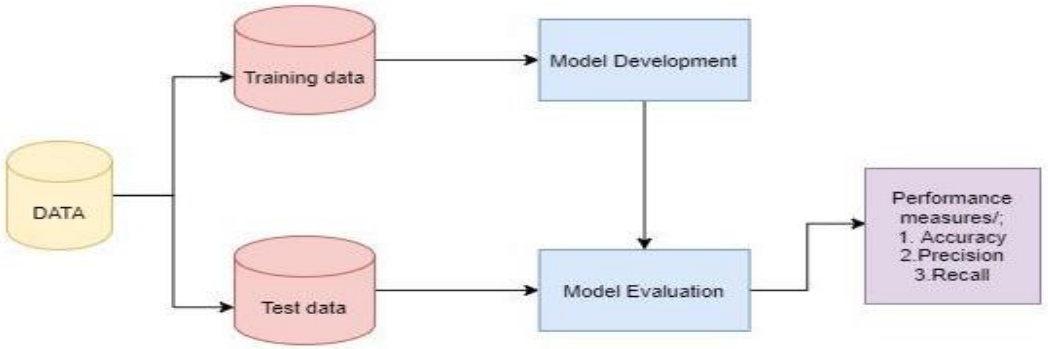
You can create an accurate or almost correct classification model even with a small amount of test data.

Your categorization results should all fall into the same category.

For example, if your collection of classes includes Math, Science, and History, your outcomes should only be Math, Science, or History, not Music.

Methodology :

Flowchart:



The suggested system's design is depicted in the diagram. The reviews of various Tamil books are gathered from various online book stores and e-book sites, then organised, examined, and manually categorised as good or negative. These categorised reviews are sent into the system as training data. From the training data, the properties are extracted. Multinomial Naive Bayes is used to develop a model. Book reviews are gathered and organised to test the system. The system receives the model file as well as the testing data as input. The reviews are classified using the Naive Bayes algorithm.

Result :

The final result is obtained and shown below,

	precision	recall	f1-score	support
negative	1.00	0.10	0.18	10
neutral	0.00	0.00	0.00	3
positive	0.86	1.00	0.92	73
accuracy			0.86	86
macro avg	0.62	0.37	0.37	86
weighted avg	0.85	0.86	0.81	86

How to improve the same in future:

In the last decade, sentiment analysis has become a popular idea in a variety of fields of study. Many scholars are working to develop a system for predicting sentiment analysis in a variety of domains. Microblogs, blogs, Facebook, Twitter, and other social media sites are popular data sources that collect relevant information. From the literature study, it can be concluded that for sentiment analysis of bigger datasets made to be accurate and efficient we need to make use of the distributed approach.

In addition, research activity may be used in a logical order to get better outcomes in this subject. Many additional feature selection approaches can be investigated in the future. The additional study can be carried out to assess the impact of a variety of regions and domain-based elements. Extending the use of sentiment analysis to other domains might provide some intriguing results. More n-grams and feature allowances can be employed in the future, resulting in higher accuracy than the existing ones. The focus of this study is on identifying qualities that exist in the form of nouns or nominal phrases in the assessments. The implicit feature analysis is saved for further use. Because ensemble learning procedures need a lot of calculation time, parallel computing methods can be used to solve this problem. The lack of outcome portrayal is a major limitation of ensemble learning systems, and the information received by ensembles is difficult to comprehend. As a result, improving ensemble interpretability might be an important research direction. Future opinion analysis structures will require a larger and more detailed general knowledge base, which will allow for better treatment of natural language views and a more effective connection between multimodal and machine-sensible data. Additional bio-inspired methods to design intelligent opinion analysis structures capable of handling semantic information, analogy creation, learning actual information, and perceiving, identifying, and sensing emotions will result from combining scientific theories about emotion with applied technology targets to analyse sentiments of natural language script.

Conclusion:

We'll end our paper with a quote: "If there is an algorithm that characterises human behaviour, emotional analysis is unnecessary." Our major objective in this research is to attain accuracy in determining an individual's positive and negative attitudes, for which we have gotten results via text mining. The sentimental approach gives a simple concept of how to analyse a certain individual's behaviour and make a judgement in a few stages to reflect that individual's perspective. Humans are now free to publish or remark on every issue that arises anywhere in the globe, thanks to the power of social networking. There may be certain ideas that aid in the development and evolution of the community or even the country. It is quite tough to extract such useful ideas from unstructured and random data and therefore swiftly arrive at a choice.

Reference:

1. Hutto CJ, Gilbert E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. Eighth International AAAI Conference on Weblogs and Social Media; 2014 May 16.

2. Fink CR, Chou DS, Kopecky JJ, Llorens AJ. Coarse- and fine-grained sentiment analysis of social media text. Johns Hopkins APL Technical Digest. 2011 Jan; 30(1):22–30.

3. Selvan A, Anand Kumar M, Soman, KP. Sentiment analysis of Tamil movie reviews via feature frequency count. IEEE International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS 15); 2015.

4. Pandey P, Govilkar S. A framework for sentiment analysis in Hindi using HSWN. International Journal of Computer Applications. 2015 Jan; 119(19):23–6.

5. Kumar A, Kohail S, Ekbal A, Biemann C. IIT-TUDA: System for sentiment analysis in Indian languages using lexical acquisition. Mining Intelligence and Knowledge Exploration. 2015 Dec; 9468:684–93.

6. Balamurali AR. Cross-lingual sentiment analysis for Indian languages using linked wordnets. CiteSeer. 2012.

7. Mittal N, Agarwal B, Chouhan G, Bania N, Pareek P. Sentiment analysis of Hindi review based on negation and discourse relation. Proceedings of International Joint Conference on Natural Language Processing; 2013. p. 45–50.

8. Timmaraju A, Khanna V. Sentiment analysis on movie reviews using recursive and recurrent neural network architectures. Semantic Scholar; 2015.

9. Anand Kumar M, Dhanalakshmi V, Soman KP, Rajendran S. Factored statistical machine translation system for English to Tamil language. Pertanika Journal of Social Science and Humanities. 2014; 22(4):1045–61.

10. Anand Kumar M, Dhanalakshmi V. A novel approach to morphological analysis for Tamil language. Germany: University of Koeln Koln; 2009 Oct.

11. Anand Kumar M, Soman KP. AMRITA-CEN@FIRE-2014: Morpheme extraction and lemmatization for Tamil using machine learning. ACM International Conference Proceeding Series; 2014 Dec. p. 112–20.

12. Anand Kumar M, Rajendran, S, Soman KP. Tamil word sense disambiguation using Support Vector Machines with rich features. International Journal of Applied Engineering Research. 2014; 9(20):7609–20.

13. Abinaya, N, Anand Kumar M, Soman KP. Randomized kernel approach for named entity recognition in Tamil. Indian Journal of Science and Technology. 2015; 8(24).

Recognition of ancient Tamil inscriptions using Convolution Neural Network

Bharathiraja A, Research Scholar, Bharathidasan University, Trichy
Gunasundari C, Research Scholar, Anna University, Chennai

Abstract -

Tamil is one of the oldest language in the world and recognized as a classical language in India, Long living Tamil literature Tholkappiyam, being the extant Tamil grammar. According to the Archaeological survey of India (ASI), Tamil language has the maximum number of inscriptions than other Indian languages, and the script has changed over the period of time and the modern tamil script has completely changed from ancient scripts. This paper proposes computer aided techniques to recognize ancient tamil characters with the help of convolution neural network. Deep learning in Artificial Intelligence(AI) has emerged in the recent decades due to availability of data and computation capability. It is time to decode ancient Tamil scripts and inscriptions by leveraging modern computer science as an epigraphist. Deep learning techniques outperforms excellent results with image, audio, video recognition and generative techniques in recent years. Tamil script has a vast history of changes in the last twenty centuries. In every century, there has been a change in scripts which is challenging even for the experts to read and recognize the script. Leveraging technology definitely would help both experts and common man to understand the scripts most correctly. The key work of this paper is to recognize the ancient tamil vowels which had different writing forms over the period of change.

Keywords - Convolutional Neural Network, Recognition of Tamil Inscription, Deep Learning

1. INTRODUCTION

1.1. Tamil language

Tamil is one of the ancient literary language and widely spoken by the Tamil people in India, Singapore, Malaysia and Srilanka. Tamil is an official language of the sovereign nations of Sri Lanka and Singapore, the Indian state of Tamil Nadu, and Puducherry[1].The earliest literary works in the Dravidian languages were in Tamil.

1.2. Ancient Tamil Script

In ancient days, Tamil ancient scripts written in Palm-leaf [olai suvadi, In Tamil:ஓலைச்சுவடி], Hero Stone [veerakkal also known as nadukal, In Tamil:நடுகல் (அ) வீரக்கல்], Inscription [kalvettu, In Tamil:கல்வெட்டு] and Copper-plate inscription [seppedu In Tamil:செப்பேடு][2]. Of these, no traces of prehistoric scripts are available; Only recent several hundred years old inscriptions have been found. Among them, we call the scripts used to write the Tamil language as Tamil [in Tamil:தமிழ்], Tamizhi [in Tamil:தமிழி], Tamil Vattezhuthu [in Tamil: தமிழ் வட்டெழுத்து].

In the third century (BC 3), king Ashoka ruled in northern India, During the reign of Ashoka, the Brahmi script was used there and the cave inscriptions in Tamil land dated back to the same period of Ashoka reign, Cave inscriptions are very helpful in understanding the ancient script of the Tamil language. There are two opinions on the naming of the written forms in these inscriptions. A few refer to it as Tamizhi[in Tamil:**தமிழி**]. Others refer to it as Tamil-Brahmi. Iravatham Mahadevan who is known for his decipherment of Tamil-Brahmi inscriptions and expertise on the epigraphy of the indus valley civilisation, he calls it as Tamil-Brahmi [in Tamil: **தமிழ் பிராமி**][3].

It is customary and believed to assume that any linguistic writing form is the same as the pictorial writing form in the early days. There is not enough evidence to know the evolution of the Tamil script, we have studied before from the beginning. Similarly, the next stage of development of Tamil writing could not be known. Table 1.1, Table 1.2, Table 1.3 illustrates the Tamil brahmi scripts and modern Tamil scripts, these examples have vowels, consonants and vowel-consonants scripts. Tamil language is an *abugida* segmental writing system which is a sequence of writing consonants and vowels in a single unit.

Modern Tamil Scripts (Vowels) **உயிர் எழுத்துக்கள்**	**அ ஆ இ ஈ உ ஊ எ ஏ ஐ ஒ ஓ**
Tamizhi or Tamil-Brahmi scripts	

Table1.1. Tamil Brami and Modern Tamil scripts (vowels)

Modern Tamil Scripts (consonants) **உயிர் எழுத்துக்கள்**	**க் ங் ச் ஞ் ட் ந் த் ண் ப் ம் ய் ர் ல் வ் ழ் ள் ற் ன்**
Tamizhi or Tamil-Brahmi scripts	

Table1.2. Tamil Brami and Modern Tamil scripts(consonants)

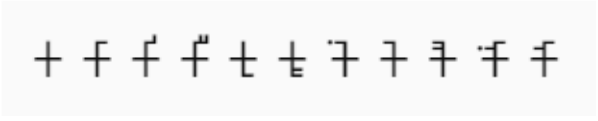
Modern Tamil Scripts **உயிர்மெய் எழுத்துக்கள்**	**க கா கி கீ கு கூ கெ கே கை கொ கோ**
Tamizhi or Tamil-Brahmi scripts	

Table1.3. Tamil Brami and Modern Tamil scripts(vowel-consonants)

1.3. Convolutional Neural Network (CNN)

Convolution Neural Network (CNN) is an architectural approach in neural networks, The structure of convolutional neural network combined with convolution, activation, and pooling operations and extended to fully connected layer followed by classification which is a softmax layer. Convolutional neural network (CNN) is inspired by the human biological neural process of. multilayer perceptron[4].

2. LITERATURE SURVEY

The prototype proposal in “Conversion of Early Tamil Brahmi Characters into Modern Tamil Characters Using Template Matching Algorithm" authored by Mageswaran et al. [7] unblock certain grey areas in character recognition of an ancient scripts, this uses various techniques such as acquisition, threshold, clustering and filtering, the key approach used here is template matching approach which is a referring point in 3-D spatial region followed by convolution neural network. This prototype has an achievement recognition of characters belonging to AD 2 to AD3[5].

Subadivya S et al, International Journal of Computer Science and Mobile Computing, Vol.9 Issue.6, June-2020, “Tamil-Brahmi Script Character Recognition system using Deep Learning Technique” [6], The proposed system uses regular preprocessing techniques followed by CNN and Dataset has been created manually, it resulted 94% accuracy.

Lalitha Giridhar, Aishwarya Dharani, Velmathi Guruviah, "A Novel Approach to OCR using Image Recognition based Classification for Ancient Tamil Inscriptions in Temples"[7], this work has a good attempt to decipher the ancient Tamil scripts between 7 to12th centuries, involves multi skills and multi layered approach from reading script to delivering as an audio file. Again the challenge is dataset creation which is done manually using the Slicer library from Python programming language.

S.Rajakumar, .V.Subbiah Bharathi, "Century Identification and Recognition of Ancient Tamil Character Recognition",International Journal of Computer Applications, This paper attempt to identify the century of the Tamil character being used and translate the ancient scripts to readable modern script, fuzzy filter and segmentation based clustering is being used for the classification technique[8].

M. Merline Magrina, "Convolution Neural Network based Ancient Tamil Character Recognition from Epigraphical Inscriptions", proposed a system that is well organized in identifying the Tamil ancient character of the 12th century. A simple CNN architecture method is used for performance and to reduce the computational cost. Bounding box method used to segregate the character and noise removal methods and other preprocessing techniques used to classify the character[9].

3. MATERIALS AND METHODS

3.1. Dataset

It is well known, the dataset for ancient script ise challenging and not available in complete and clean set in any of the forums, but the papers being published on character recognition on Tamil ancient scripts have used the dataset which are created by individuals of authors. Most of the research work on ancient script recognition refers to Tamil-Brahmi letters which are dated between 3 BC to 3 AD. Our research work is trying to recognize all the ancient letters found which are prior to modern character. Ancient scripts for the last 20 centuries given in Table 3.1.[10]

Century	**BC3**	**AD2**	**AD3**	**AD4**	**AD5**	**AD6**	**AD7**	**AD8**	**AD9**	**AD10**
அ										
Century	**AD11**	**AD12**	**AD13**	**AD14**	**AD15**	**AD16**	**AD17**	**AD18**	**AD19**	**AD20**
அ										

Table 3.1. Tamil ancient script writing method of the letter 'அ' for 2000+ years

Tamil vowels have changed and evolved every century to reach modern characters which are in use now. In order to recognize the character irrespective of the century, the model to recognize and understand the letters used in different timelines is the same.

3.2. Proposed system

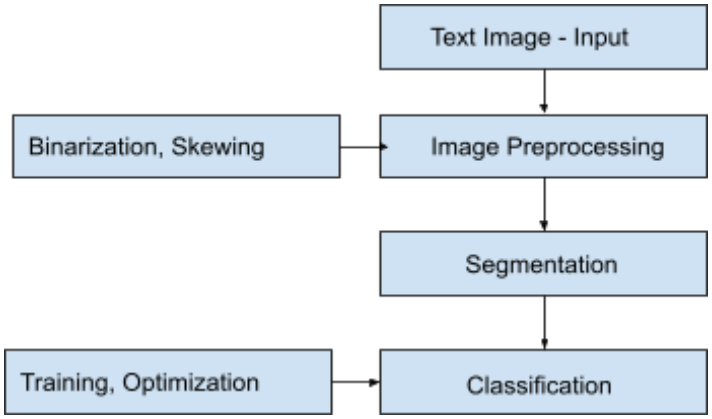


Figure 3.1 Proposed System

The proposed system has been implemented with various techniques in preprocessing that removes noise followed by segmentation to split the characters from the texture script, lastly classification with manual accumulation on top of softmax computation. Figure 3.1 illustrates the proposed system that has been discussed in the subsequent sections of this paper.

3.3. Preprocessing Techniques

The object of preprocessing technique helps to distinguish the character from the source image and the source image could be in any form such as stone inscriptions, palm writings etc.. Extracting characters from the sources of centuries is really difficult, we use few preprocessing techniques to overcome the challenges.

3.3.1. Binarization

Binarization techniques can be used in image preprocessing in order to recognize or detect the input image. Binarization works based on thresholding methods. The most popular binarization technique is Local binarization that calculates for each image pixel the mean and the standard deviation of gray scale value of the neighboring pixels[13]. Threshold is defined as T,

$$\text{Threshold } T = \text{Mean} + k * \text{Stdev}$$

Where Mean is average value of the pixel in the local area, stdev is the standard deviation of the same pixel, and k is a constant

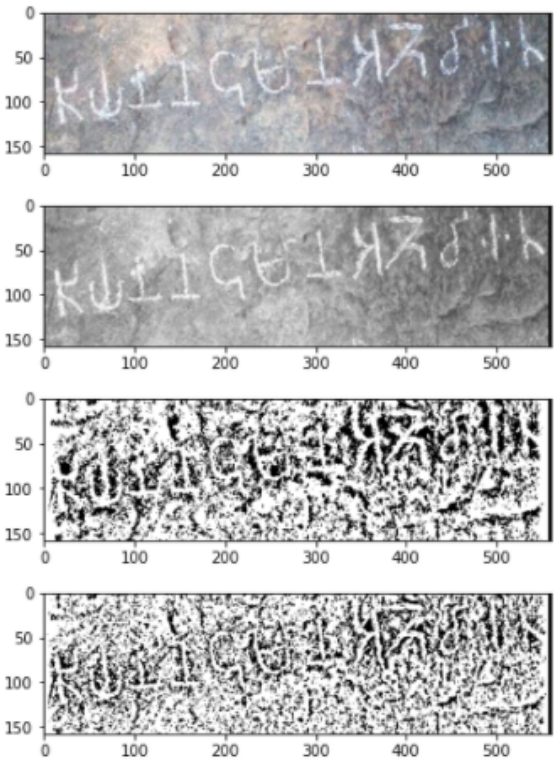


Figure 3.2. a) Original Image b) GrayScale c) Mean d) Gaussian

3.3.2 Skew correction

Inscriptions are handwritten and the alignment of the texture changes also the surface where inscriptions are written would not be an even space, some of the characters would give different translation if tilted slightly, thus the skew correction is required.

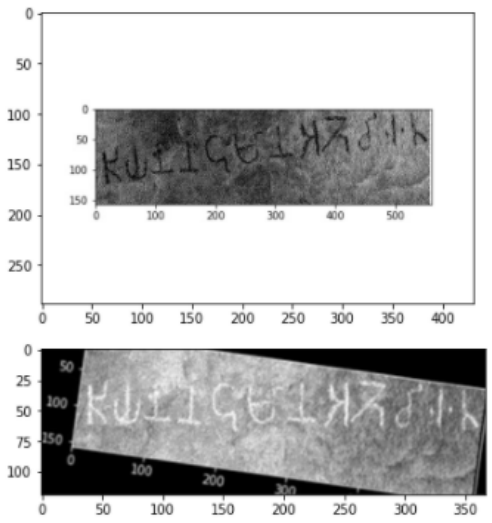


Figure 3.3. Skew correction

3.4. Segmentation

Segmentation of character would be the series of processes which goes by the order of segment the lines, segment the words and segment the characters. Histogram projection methods use content frequency plot to differentiate the characters. Vertical and Horizontal histogram used for the segmentation techniques.

3.5. System Architecture

CNN also known as convolutional Neural Network is used for classifying the ancient tamil inscription characters. In general, convolutional neural networks consist of convolutional and pooling, and fully connected layers.A convolutional layer is used for feature extraction from a given input image with spatial information, therefore CNN preserves the relationship among pixels. Convolutional layer uses certain properties such as Filter or Kernel which is a feature map, Stride and Padding. Pooling layer is used for image compression, Pooling layer uses max pooling. The fully connected layer converts the matrix information to a vector which is given to softmax for classification. CNN architecture for the proposed system is in Figure 3.4. CNN is used mainly for not losing the spatial property which produces higher accuracy than a fully connected layer which is a highly complex model. CNN performed well in the last two decades in image processing techniques and won in contests for the outstanding performance in recent years.

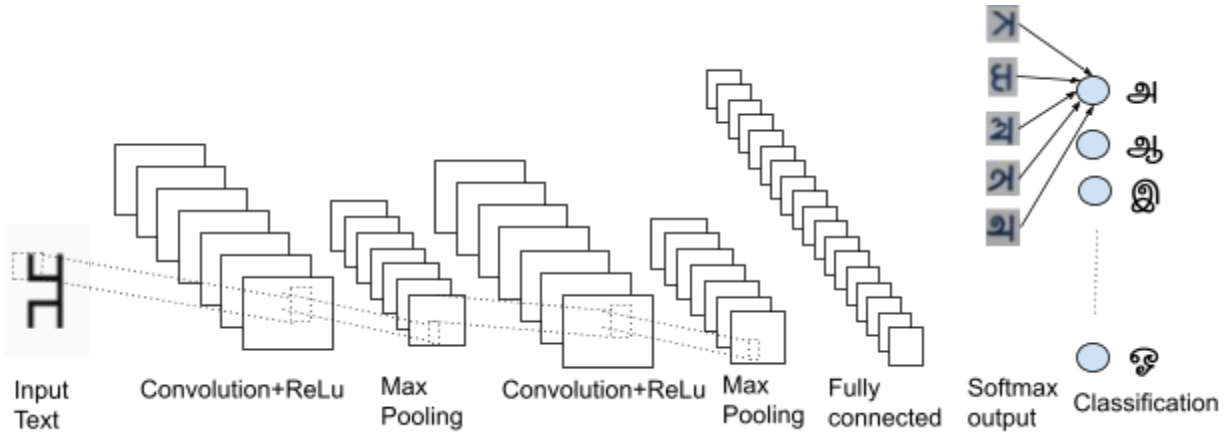


Figure 3.4. Proposed CNN Architecture

The application of convolutional layer convolves an input image with kernels to get a feature map,Unit in a feature map is connected to the previous layer through the kernel's weight and the weights of the kernels are optimized for best during the training by backpropagation method, non-linear activation(ReLu) will be applied on each neural output and the Rectifier linear units (ReLU) represented as[4] ,

$$f(x) = \max(0,x)$$

Rectifier linear units (ReLU) became a most popular and faster activation function as of 2017 for deep neural networks.Comparatively, ReLu is faster than sigmoid and tanh activation functions. Fully connected layer will be used once the convolution operation is completed, meaning the matrix will be converted to a vector and it will be sent to the fully connected layer. Pooling is a strategy for downsampling the image with spatial information, Figure 3.5. demonstrates max pooling of 4X4 matrices.

21	1	3	11
2	32	6	10
4	7	23	12
3	9	11	43

32	11
9	43

Figure 3.5. max pooling of 4X4 matrix

Lets see some of the additional techniques used along with CNN in order to optimize the model for high accuracy such as padding, Batchnomalization, dropouts. Image padding introduces additional pixels around four sides of the matrix and it is used to detect spatial structures in the boundary of the image. The major advantage of CNN compared to traditional machine learning models is to detect features automatically without any human intervention whereas traditional models have a person intervention in feature extraction meaning CNN can learn features by itself from the given pictures of any.

3.6. Proposed Convolutional Neural Network and Implementation Details

The Convolutional Neural Network (CNN) architecture is shown in Table 3.1, In this paper, we propose a fixed input size 32x32 pixels and 3 channels. We used the first convolution with 3x3 kernel size and 32 filters, followed by a max pooling with 2x2 pixels, similarly the second convolution with 3x3 kernel size and 64 filters followed by a max pooling 2x2 pixels, Rectified Linear unit (ReLu) used as a activation function in this architecture. Optimizer used to modify the CNN attributes such as weight and learning rate, Adam is used as optimizer in the proposed system.Also we used sparse categorical cross entropy which will match the most likely similar or matching category whereas categorical cross entropy uses on-hot matching. Dense layers used in last layers for classification.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_1 (Conv2D)	(None, 13, 13, 64)	18496
max_pooling2d_1 (MaxPooling2	(None, 6, 6, 64)	0
conv2d_2 (Conv2D)	(None, 4, 4, 64)	36928
flatten (Flatten)	(None, 1024)	0
dense (Dense)	(None, 64)	65600
dense_1 (Dense)	(None, 7)	455

Total params & Trainable params : 122,375

Table 3.1. Proposed-CNN architecture model summary

3.7. Experimental Setup and Result Analysis

As explained in the previous section, the model has been trained with 10 epochs, that gives more than 90% accuracy, and an increase in epoch results in more accuracy with the given model, with the above model the results are observed as - *loss: 0.2190 - accuracy: 0.9266 - val_loss: 0.2201 - val_accuracy: 0.8873*, Figure 3.6 shown the comparison plot between training and validation results. Since the difference between training and validation results shows less difference in accuracy and loss parameters,

the model could be most appropriate.. In addition to accuracy and loss, there are other parameters such as performance and confusion matrix resulting in better performance. On an average each epoch takes 30ms to 50 milliseconds. Figure 3.7 shows accuracy and loss tensorLog during the training of the model. Python programming language and tensorflow framework have been used for this implementation and results. The overall test loss and test accuracy as given below based on the model training in Figure 3.7.

Test loss: 0.23422138392925262

Test accuracy: 0.9428571462631226

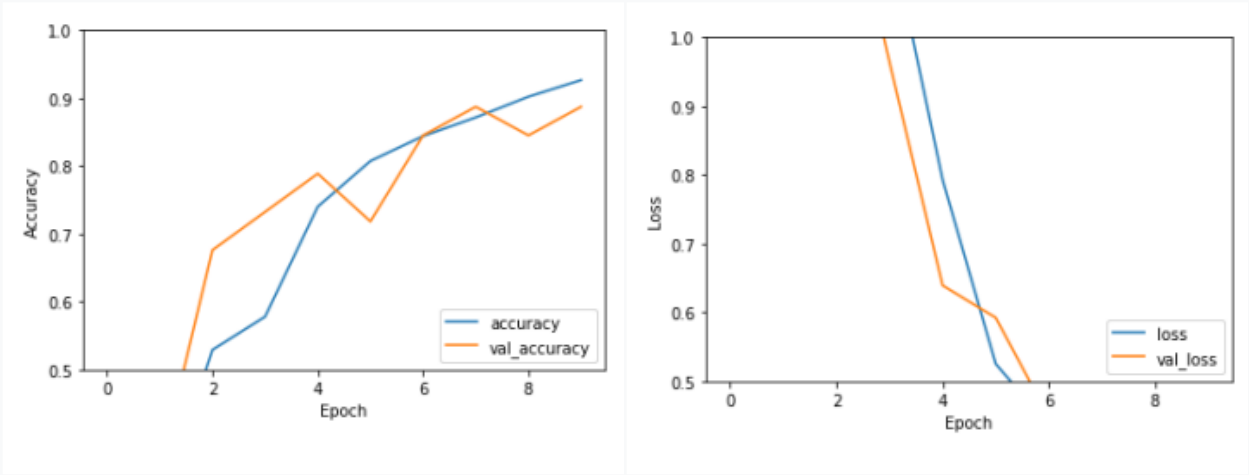


Figure 3.6. Result comparison between Accuracy & validation accuracy and Result comparison between loss & validation loss

Epoch 1/10
11/11 [==============================] - 1s 61ms/step - loss: 1.7456 - accuracy: 0.3242 - val_loss: 1.6701 - val_accuracy: 0.0845
Epoch 2/10
11/11 [==============================] - 0s 38ms/step - loss: 1.5464 - accuracy: 0.2905 - val_loss: 1.5297 - val_accuracy: 0.3521
Epoch 3/10
11/11 [==============================] - 0s 40ms/step - loss: 1.4831 - accuracy: 0.5291 - val_loss: 1.2968 - val_accuracy: 0.6761
Epoch 4/10
11/11 [==============================] - 0s 40ms/step - loss: 1.1590 - accuracy: 0.5780 - val_loss: 0.9640 - val_accuracy: 0.7324
Epoch 5/10
11/11 [==============================] - 0s 41ms/step - loss: 0.7929 - accuracy: 0.7401 - val_loss: 0.6391 - val_accuracy: 0.7887
Epoch 6/10
11/11 [==============================] - 0s 39ms/step - loss: 0.5250 - accuracy: 0.8073 - val_loss: 0.5922 - val_accuracy: 0.7183
Epoch 7/10
11/11 [==============================] - 0s 40ms/step - loss: 0.4352 - accuracy: 0.8440 - val_loss: 0.4474 - val_accuracy: 0.8451
Epoch 8/10
11/11 [==============================] - 0s 40ms/step - loss: 0.3212 - accuracy: 0.8716 - val_loss: 0.3479 - val_accuracy: 0.8873
Epoch 9/10
11/11 [==============================] - 0s 40ms/step - loss: 0.2877 - accuracy: 0.9021 - val_loss: 0.3763 - val_accuracy: 0.8451
Epoch 10/10
11/11 [==============================] - 0s 40ms/step - loss: 0.2190 - accuracy: 0.9266 - val_loss: 0.2201 - val_accuracy: 0.8873

Figure 3.7 . The TensorLog during training the model showing loss and accuracy

3.8. Comparison with other methods

Comparison of similar work, that is recognition of ancient tamil scripts shown in Table 3.2, Every century has its own style of character, but the work proposed in the papers have different parameters, and it cannot be compared directly thus the method and dataset used have been provided for the high level comparison. Every work given the table 3.2 has significant improvement based on the parameters considered.

Authors	Methods/Dataset	Accuracy
S. Mageshwaran et al. [5]	Template Matching Approach, CNN /Tamil-Brahmi characters of between AD2 and AD3	-
Subadivya S et al. [6]	CNN, Natural Language Processing	94.6%
Lalitha Giridhar et al. [7]	OCR,CNN, Google translator	77.7%
S.Rajakumar et. al [8]	Fuzzy median filters, Neural network, Clustering	-
M. Merline Magrina et al. [9]	Bouncing Box character Recognition, CNN / Recognition of 12 AD characters	98.61%
Our Approach	CNN, Softmax accumulation / Manually created data set for the centuries from BC3 - AD19, Tamil Vowels	92.66%

Table 3.2. Comparison summary with similar work

Our approach uses the dataset for twenty plus years of century characters which has a change in every century. By combining the results, the model is classifying the characters if there is a similarity in scripting.

3.9. Extension of softmax layer

The softmax layer is a classification layer in Convolutional Neural Network that turns the vector real value into a vector of values whose probability distribution sum upto the value 1. The example of probability distribution given as follows for letter 'அ', and the computation of softmax value is 1. highest value is considered as classified result,

$$\sigma(\vec{z})i = \frac{e^{zi}}{\sum_{j=1}^{k} e^{z_j}}$$

Where z is input vector to the softmax function
K is number of classes in multi-classification

In the proposed system we classify the character of 2000 years of characters that varies in every century. We grouped similar strokes of the vowel 'அ' into 6 different classes, instead of making it as a single class as classified in table 3.2. Sum of sub classes of parent vowels are considered for classification which

results in maximum accuracy and it reaches more than 90%, by adding more dataset would help to increase the accuracy, in this case for the vowel 'அ', we created 6 subclasses.This approach slightly increases the computation time Recognition of vowels resulting in 12 classes but the softmax layer will have more than 12 classes which are subclasses of parent classes. The classification of softmax grouped computation is given in Table 3.3.

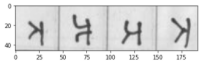
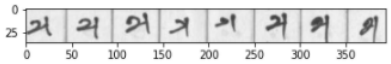
Class 1 (BC 3 to AD 4)	
Class 2 (AD 5)	
Class 3 (AD 6)	
Class 4 (AD 7)	
Class 5 (AD 8 to AD 12)	
Class 6 (AD 13 to AD 20)	

Table 3.2. Grouping of similar character set

4. CONCLUSION AND FUTURE WORK

In this paper, character recognition of Tamil ancient script is demonstrated. The proposed method to classify the ancient tamil character using Convolutional Neural Network with extended grouped classes technique that helps softmax accumulation to increase the accuracy of classification. Softmax accumulation can be used as optional and it depends on the accuracy and helps to break the tie if any. Thus the accuracy increases significantly than other methods which deal with single classes for a character. Also proposed experiment trained and tested with manually created dataset and only with vowels, we are further inspired to think and extend this work to consonants and vowels-consonants along with actual inscriptions.

5. REFERENCE

1. Wikipedia,[online] https://en.wikipedia.org/wiki/Tamil_language
2. Tamil Virtual University, [online] http://www.tamilvu.org/courses/degree/a051/a0514/html/a051414.htm
3. Iravatham Mahadevan, [Text Book] "Early Tamil Epigraphy from the Earliest Times to the Sixth Century A.D"

4. Sakshi Indolia, Anil Kumar Goswami, S.P.Mishra Pooja Asopaa, "Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach",Procedia Computer Science, Volume 132, 2018, Pages 679-688
5. S. Mageshwaran, C. Ravishankar, G. Alagumalaikannan, M. Kavin, S. S. L. DuraiArumugam, "Conversion of Early Tamizh Brahmi Characters into Modern Tamil Characters Using Template Matching Algorithm",ISSN: 2278-0181, 2018
6. Subadivya S, Vigneswari J, Yaminie M, Diviya M, "TAMIL-BRAHMI SCRIPT CHARACTER RECOGNITION SYSTEM USING DEEP LEARNING TECHNIQUE", IJCSMC, Vol. 9, Issue. 6, June 2020, pg.114 – 119, ISSN 2320–088X, 2020
7. Lalitha Giridhar, Aishwarya Dharani, Velmathi Guruviah, "A Novel Approach to OCR using Image Recognition based Classification for Ancient Tamil Inscriptions in Temples"
8. S.Rajakumar, .V.Subbiah Bharathi, "Century Identification and Recognition of Ancient Tamil Character Recognition",International Journal of Computer Applications (0975 – 8887), Volume 26– No.4, July 2011
9. M. Merline Magrina, "Convolution Neural Network based Ancient Tamil Character Recognition from Epigraphical Inscriptions", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056,p-ISSN: 2395-0072, Apr 2020
10. [Inscription source], https://sites.google.com/site/msvkgf/tamils-scripts
11. Boiangiu, C. - A.; Olteanu, A.; Stefanescu, A. V.; Rosner, D. & Egner, A. I. (2010). Local Thresholding Image Binarization using Variable-Window Standard Deviation Response (2010). 0133-0135, Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, ISBN 978-3-901509-73-5, ISSN 1726-9679, pp 0067, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria 2010
12. N Sasipriyaa et al., "Handwritten Ancient Tamil Character Recognition Using Generative AdversarialNetwork", Turkish Online Journal of Qualitative Inquiry (TOJQI), Volume 12, Issue 3, July 2021:1849- 1857

Translingual Architecture for Prediction of 2021 Tamil Nadu State Assembly Elections

[1*]Ferdin Joe John Joseph, [2]Ganapathy Sannasi

[1]Faculty of Information Technology, Thai-Nichi Institute of Technology, Bangkok, Thailand

[2]School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, India

[*]Corresponding Author: ferdin@tni.ac.th

Abstract:

Sentiment Analysis is gaining popularity in mood mapping the people using the advances of Artificial Intelligence. Many elections are being predicted using the texts collected the micro blogging site twitter. Decision Tree based architectures were tried on 2019 Indian General Election and 2020 Delhi Election predictions. When a preliminary study was done to predict the Tamil Nadu State Assembly elections held in May 2021, the data collected indicated a need for translingual support for predicting the sentiment of people during the course of poll campaign. Since this election is predominantly revolving around a majority Tamil speaking heartland, a new architecture was developed and analyzed. The results using a proposed translingual architecture predicted was close to the actual results and the statistical insights to prove this method shows the evidence that this method is better than qualitative and quantitative surveys done by media for pre poll and exit poll prediction.

Keywords: Sentiment Analysis, Election Prediction, Indian Election, Opinion Mining

Introduction

Tamil Nadu State Assembly Elections are conducted once in five years since the independence of the Republic of India. It was conducted as State Assembly polls of erstwhile Madras presidency until 1967 and Tamil Nadu until the recent 2021 elections. In the current scenario, there are 234 constituencies where the party gaining support of at least 118 Members of Legislative Assembly will reserve the right and get invited to form the Government. Post-independence, the state has faced elections in 1952, 57, 62, 67, 72, 77, 80, 84, 89, 91, 96, 2001, 06, 11, 16 and 21. Prediction of election outcomes through sample surveys are usually done with pre-poll and exit poll scenarios. In the recent years, since the election prediction of Nate Silver on 2012 US presidential polls (Gelman et al., 2012), many AI based prediction methods were used to predict the elections in various countries over the past decade. Sentiment Analysis using various machine learning algorithms used. Social media impact by microblogging sites are a prominent factor in effecting the thought process of people (F. J. John Joseph, 2011). The reverse of this is meant as mood mapping of people or identifying the collective data from people. This mood mapping coupled with sentiment analysis on the text posted in microblogging sites are helpful in predicting the outcome of elections. This has been done in many countries like USA (Yang et al., 2018), Australia (Burgess & Bruns, 2012), India (F. J. J. John Joseph, 2019), Indonesia (Budiharto & Meiliana, 2018) and states like Delhi (F. J. John Joseph & Nonsiri, 2021). AI based methods (Graphnile, 2020) are also used by some companies to provide poll analytics.

Related Work

Tamil Computing has been increasing its fame over the last couple of decades with advances in Offline Character Recognition (OCR) and Natural Language Processing (NLP) especially (J et al., 2010). An extensive pool of research has been done on NLP with English. For languages other than English, there are some manual labelling-based approaches.

A popular opinion based method was used in Queensland state elections in Australia (Burgess & Bruns, 2012). An opinion mining based approach was developed for 2013 Pakistani General Elections and 2014 Indian general elections (Kagan et al., 2015). Decision Tree was used in predicting 2019 Indian General Elections (F. J. J. John Joseph, 2019) and region based opinion mining for Delhi Assembly elections in 2020 (F. J. John Joseph & Nonsiri, 2021). A clustering was done on 6000 tweets in US presidential polls 2016 between Donald Trump and Hillary Clinton (Ramteke et al., 2016). Swedish election was predicted using support vector machines (Larsson & Moe, 2012). Apart from these research, Graphnile is doing AI based political analysis and started predicting elections since 2019 Indian General Elections (Graphnile, 2020).

Methodology

A modified methodology as done in (F. J. John Joseph & Nonsiri, 2021) is added with the a python translator. The following steps are followed to predict the outcome of election conducted in the state of Tamil Nadu in May 2021.

Data Collection

Tweets were collected during the mid of march and the preliminary analysis were done on the type of users pooling in through the API (Roesslein, 2009). The distribution of tweets based on languages, devices and location are listed in figure 1 below. This calculation is done on an average figure obtained from a continuous collection of data within a range of days.

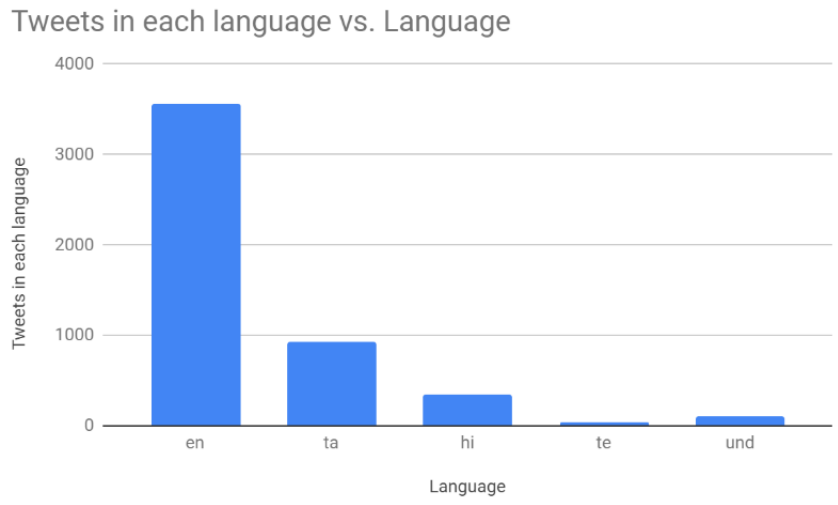


Figure 1: Distribution of tweets based on Languages

The tweets were found to be more English or anglicized texts of other languages. There is a neat trend which indicates that the ratio of Tamil tweets against English texts is higher than the ratio of Hindi tweets against English tweets as observed during Delhi state assembly polls in 2020 (F. J. John Joseph & Nonsiri,

2021). The correlation matrix of subjectivity obtained on a sample set of tweets is given in the figure 2 below:

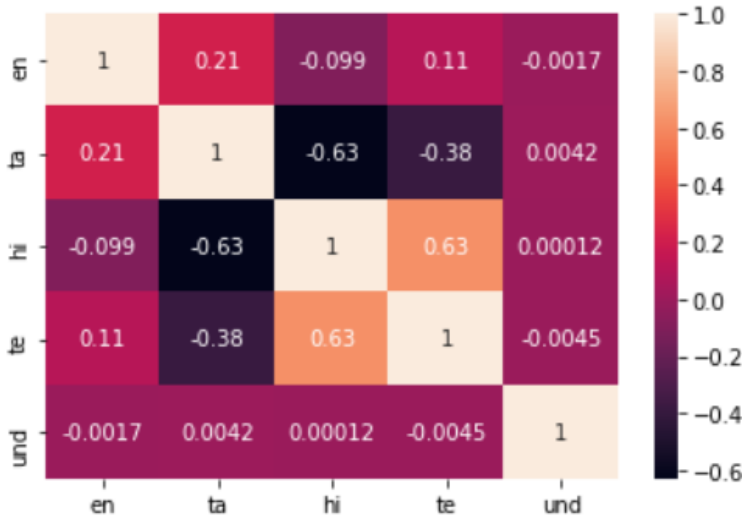


Figure 2: Correlation matrix of polarities obtained from major languages

Architecture

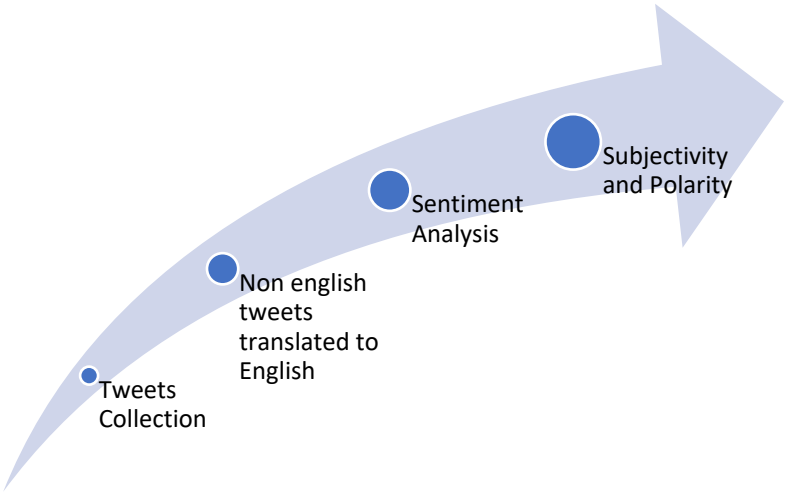


Figure 3: Translingual Architecture for tweet analysis

Owing to the positive correlation between the polarities of Tamil with English tweets, the translator engine is added to convert the Tamil tweets and then calculated for the popularity. This is done using the weighted decision tree. Translation to English will facilitate the classification of tweets using the NLTK corpus (Loper & Bird, 2002). The polarity is taken to extrapolate the results as seats. This is calculated against similar decision tree-based approaches and the actual prepoll analysis and exit poll predictions.

With the polarity trend obtained in the Figure 4, the data is extrapolated on the district wise trends with respect to the tweets within those latitudes and the count is estimated for the possible number of seats to be won by the political alliances. This has been done for the experimentation in all the AI based methods.

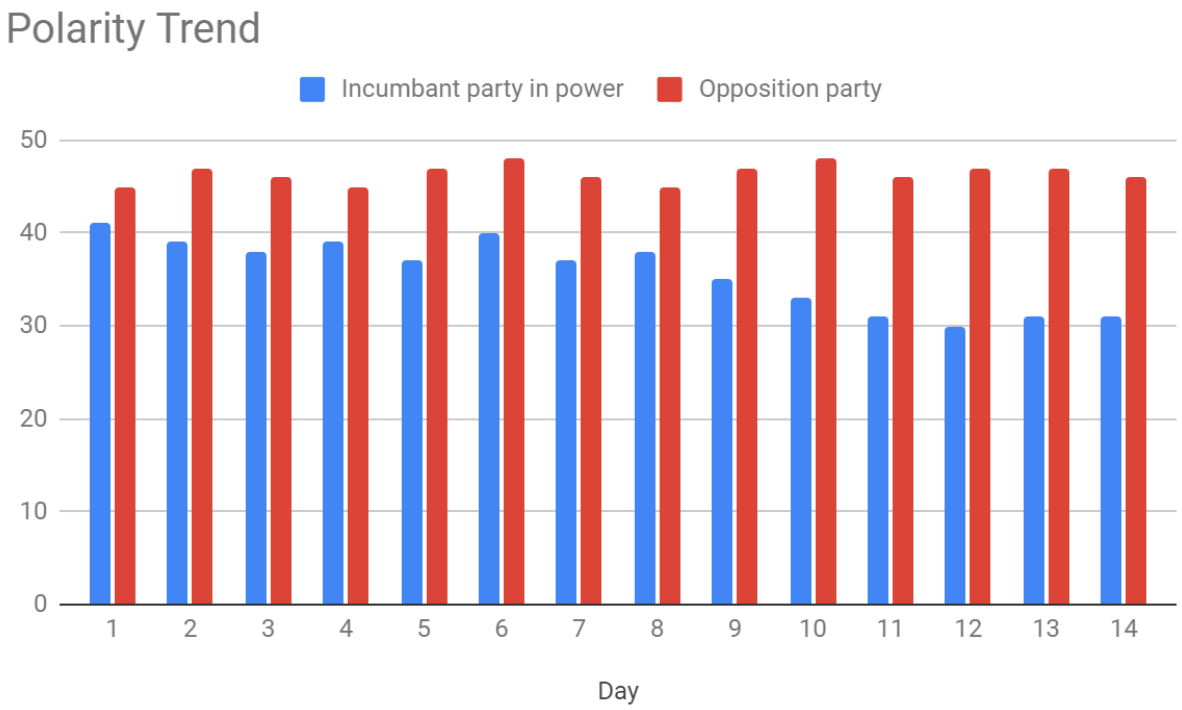


Figure 4: Polarity Trend

Table 1: Performance comparison

S.No	Approach	Methodogy	Predicted seats for winning party
1.	Pre poll	Times Now - CVoter	177
2.		ABP News - CVoter	166
3.	Exit Poll	Republic TV - CVoter	160
4.		Spick News	146
5.	AI	(F. J. J. John Joseph, 2019)	132
6		(F. J. John Joseph & Nonsiri, 2021)	165
7		Proposed translingual architecture	150
8	Actual	Poll results	159

The performance of the proposed translingual architecture is a bit close to the actual figures. This is due to the tweets from Tamil are also added to measure the sentiment of tweets analyzed. Despite of unavailability of raw sentiment analysis on Tamil corpus, it is converted to English translation which increased the sample of data collected and strengthens the core of opinion and mood mapping. With a specific sentiment analyzer exclusive for Tamil Language, the opinion mining on issues related to the Tamil heartland can be surveyed in a more efficient manner.

Conclusion

It is evident from the qualitative results that the performance of translingual architecture on Tamil and English tweets were found to be close enough to predict the outcome of election. However, the results are not precise enough to the levels of individual constituencies but it is a compatible performance evaluation against all the existing methods done at the levels of pre-poll, exit-poll and AI. There lies a need for the

development of Tamil NLT toolkit for classifying the sentiments from texts published online. Since the number of tweets collected are small enough for a machine learning algorithm to handle, deep learning transformers are not tried. Ultimately this is a competent methodology to arrive at the sentiment trend of Tamil texts from microblogging sites.

References

Budiharto, W., & Meiliana, M. (2018). Prediction and analysis of Indonesia Presidential election from Twitter using sentiment analysis. *Journal of Big Data*, *5*(1), 51.

Burgess, J., & Bruns, A. (2012). (Not) the Twitter election: the dynamics of the# ausvotes conversation in relation to the Australian media ecology. *Journalism Practice*, *6*(3), 384–402.

Gelman, A., Silver, N., & Edlin, A. (2012). What is the probability your vote will make a difference? *Economic Inquiry*, *50*(2), 321–326.

Graphnile. (2020). *Graphnile Election Analytics*. Online. https://www.graphnile.com/electoral-analytics

J, F. J., Ravi, T., & Velayutham, P. R. (2010). Effective Tamil Character Recognition in Tablet PCs Using Pattern Recognition. *Tamil Internet Conference*.

John Joseph, F. J. (2011). Impact of SOA and Web 2 . 0 in Tamil Blogs and Social Networks. *Proceedings - 10th Tamil Internet Conference*, 213–215.

John Joseph, F. J. J. (2019). Twitter Based Outcome Predictions of 2019 Indian General Elections Using Decision Tree. *Proceedings of 2019 4th International Conference on Information Technology*, *October*, 50–53. https://doi.org/10.1109/INCIT.2019.8911975

John Joseph, F. J., & Nonsiri, S. (2021). Region-Specific Opinion Mining from Tweets in a Mixed Political Scenario. *International Conference on Intelligent and Smart Computing in Data Analytics*, 189–195.

Kagan, V., Stevens, A., & Subrahmanian, V. S. (2015). Using twitter sentiment to forecast the 2013 pakistani election and the 2014 indian election. *IEEE Intelligent Systems*, *30*(1), 2–5.

Larsson, A. O., & Moe, H. (2012). Studying political microblogging: Twitter users in the 2010 Swedish election campaign. *New Media & Society*, *14*(5), 729–747.

Loper, E., & Bird, S. (2002). NLTK: the natural language toolkit. *ArXiv Preprint Cs/0205028*.

Ramteke, J., Shah, S., Godhia, D., & Shaikh, A. (2016). Election result prediction using Twitter sentiment analysis. *2016 International Conference on Inventive Computation Technologies (ICICT)*, *1*, 1–5.

Roesslein, J. (2009). tweepy Documentation. *Online] Http://Tweepy. Readthedocs. Io/En/V3*, *5*.

Yang, X., Macdonald, C., & Ounis, I. (2018). Using word embeddings in twitter election classification. *Information Retrieval Journal*, *21*(2–3), 183–207.

தமிழ் மொழிக்கான சொற்சூழல் கருவி

முனைவர் இரா.அகிலன்
நிரலாளர்
செம்மொழித் தமிழாய்வு மத்திய நிறுவனம்
சென்னை
akilan.rp@gmail.com
9965734497

1. அறிமுகம்

இயற்கை மொழி ஆய்வு என்பது இயற்கை மொழி அமைப்பைக் கணினிக்கு ஏற்ற வகையில் மின் இலக்கணமாகக் கொடுப்பது. கணினிக்கு இயற்கை மொழி ஆய்வை மேற்கொள்ளும் அறிவுத்திறனைக் கொடுப்பதற்குப் பல மொழிப் பயன்பாட்டுப் பணிகளை மேற்கொள்ள வேண்டும். அவற்றில் மேற்குறிப்பிட்ட தரவகத்தை உருவாக்கவும் அல்லது உருவாக்கப்பட்ட தரவகத்தை அடிப்படையாக் கொண்டு மொழிப் பயன்பாட்டுப் பணிகளுக்கான மொழிக் கருவிகள் உருவாக்கப்பட வேண்டும். இந்த மொழிப்பயன்பாட்டு மென்பொருள்களின் சொற்சூழல் கருவி என்பது அடிப்படையான ஒன்று. சொற்சூழல் கருவி என்பது கொடுக்கப்பட்ட ஒரு தரவில் அமைந்துள்ள சொற்களின் சூழலைப் புரிந்து கொள்வதற்கு ஏதுவாக அச்சொல்லின், முன் பின் உள்ள சொற்கள் ஆகியவற்றை ஓர் அடைவாகப் பயனாளருக்கு இனம் காட்டக்கூடிய கணினியின் நிரல் தொகுப்பாகும். இக்கட்டுரை தமிழ் மொழிக்கான சொற்சூழல் கருவி, தரவக உருவாக்கத்தில் சொற்சூழல் கருவியின் பங்கு, சொற்சூழல் கருவியின் பயன்பாடுகள், கருவி உருவாக்கத்தில் ஏற்படும் சிக்கல்கள் மற்றும் தீர்வுகள் பற்றி விரிவாக ஆராய்வதே இவ்வாய்வுக் கட்டுரையின் நோக்கமாக அமைகிறது.

முக்கியச் சொற்கள் : சொற்சூழல், மொழிக்கருவிகள், தரவகம், சொற்சூழல் கருவி, தமிழ்

2. தரவகம்

ஒரு குறிப்பிட்ட ஒழுங்குமுறையுடன் அதிக அளவில் தேர்வுச் செய்யப்பட்டுச் சேமிக்கப்பட்ட இயற்கையான நடைகளை உடைய உரைகள் மற்றும் பல்வேறு பனுவல்களின் தொகுப்புப் பெருந்தரவு எனப்படும். இ்ந்து ஒரு குறிப்பிட்ட மொழியின் சொற்கள் அல்லது துறைச் சொற்களை உள்ளடக்கியதாக இருக்க வேண்டும். ஒரு மொழியின் பல்வேறு பரிணாமங்களை எதிரொலிப்பதாகவும் பலதரப்பட்ட புலங்களுக்கு முதன்மை அளிப்பதாகவும் அறிவியல் முறைப்படியும் இருக்க வேண்டும். தரவகம் என்பது பல்வேறு வகைப்படும். ஒரு மொழியின் வரலாற்று வளர்ச்சியைக் காண்பதற்கான தரவகம் (Historical Corpus) என்றும், ஒரு குறிப்பிட்ட மொழியின் இன்றைய அமைப்பை அல்லது இலக்கணத்தைப் பெறுவதற்கான தரவகம் (Synchronic Corpus) என்றும், மொழிக் கற்பித்தல், கற்றலுக்கான தரவகம் (Corpus For Language Learning/Teaching), என்றும் இயந்திர மொழிபெயர்ப்புக்களுக்குப் பயன்படும் இரண்டு மொழிகள் அல்லது அதற்கு மேற்பட்ட

மொழிகளின் இணைகளை உள்ளடக்கிய தரவகம் (Parallel Corpus) என்றும் பல வகைகளில் பகுக்கப்படுகின்றன. ஆய்வாளரின் நோக்கத்திற்கேற்ப தரவகமும் உருவாக்கப்படவேண்டும். இவ்வாறு பல்வேறு புலங்களுக்குப் பயன்படும் வகையில் தரவகங்களை உருவாக்க பல்வேறு வகையான மொழியியல் ஆய்வுக் கருவிகள் பயன்படுகின்றன. இவற்றில் தரவகங்களை உருவாக்கப் பயன்படும் கருவிகளில் சொற்சூழல் கருவி இன்றியமையாதது. இக்கட்டுரையில் சொற்சூழல் கருவி வழியாகத் தரவகம் உருவாக்கம் பற்றி அறியலாம்.

3. இந்திய மொழிகளுக்கான இயற்கை மொழி ஆய்வுக் கருவிகள்

இந்திய மொழிகளுக்கான இயற்கை மொழி ஆய்வுக் கருவிகள் பல்வேறு பல்கலைக்கழகங்கள், ஆய்வு நிறுவனங்கள், தனியார் ஆர்வலர்கள் இணைந்து உருவாக்கத்தில் ஈடுபட்டு வருகின்றனர். மத்திய அரசின் தகவல் தொழில்நுட்ப அமைச்சகம் இதற்கென்ற ஒரு தனித்திட்டத்தை இந்திய மொழிகளில் தொழில் நுட்ப வளர்ச்சி (Technology Development of Indian Language – TDIL) என்ற திட்டத்தை உருவாக்கிச் செயல்பட்டு வருகிறது. தமிழகத்திலும் மாநில அரசானது தமிழ் மொழியின் தொழில்நுட்ப வளர்ச்சிக்காகப் பல்வேறு திட்டங்களைச் செயல்படுத்தி வருகிறது. மேலும், சென்னை தமிழ் இணையக் கல்விக்கழகம், செம்மொழித் தமிழாய்வு மத்திய நிறுவனம், சென்னை. ஹைதராபாத் பல்கலைக்கழகம் தெலுங்கானா, அண்ணாப் பல்கலைக்கழகம் கணிப்பொறியியல் துறை, சென்னை குரோம்பேட்டை AU-ICBC மையம், அண்ணாமலைப் பல்கலைக்கழகம் சிதம்பரம், தஞ்சை தமிழ்ப் பல்கலைக்கழகம், கோவை பாரதியார் பல்கலைக்கழகம், சென்னை SRM பல்கலைக்கழகம், இந்திய தொழில்நுட்ப கழகம் (IIT), அமிர்தா பல்கலைக்கழகம் கோயமுத்தூர், இந்திய அறிவியல் கழகம், பெங்களூரு, மைக்ரோ சாப்ட்வேர், பெங்களூர் முதலான நிறுவனங்கள் தமிழ் மொழிக்கான தொழில்நுட்பக் கருவிகள் உருவாக்கத்தில் தொடர்ந்து ஈடுபட்டு வருகின்றன.

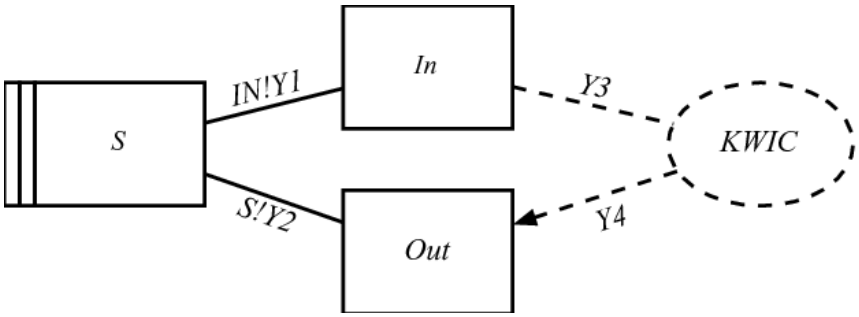
4. சொற் சூழல் கருவி

சொற்சூழல் கருவி என்பது கொடுக்கப்பட்ட ஒரு தரவுகளில் அமைந்துள்ள சொற்களின் சூழலைப் புரிந்து கொள்வதற்கு ஏதுவாக அச்சொல்லின் முன், பின் உள்ள சொற்கள் ஆகியவற்றை ஓர் அடைவாகப் உருவாக்கி அளிக்கக்கூடிய கணினி நிரல். ஒரு மொழியில் அமைந்துள்ள சொற்களின் சூழலைப் புரிந்து கொள்வதற்கு அடிப்படையான சொற்சூழல் கருவியைக் கொண்டு உருவாக்கப்பட்டுள்ள இந்த மென்பொருளில் சொல், சொல் நிகழ்வெண், முதன்மைச் சொல்லுக்கு முன் உள்ள சொற்கள், முதன்மைச் சொல்லுக்குப் பின் உள்ள சொற்கள், செய்யுள் இடம்பெற்றுள்ள மூலநூலின் அடிகள் முதலானவற்றைப் பெறுமாறு வடிவமைக்கப்பட்டுள்ளன. இஃது, ஒரு மொழியில் அமையும் சொற்களின் முன்னாகவும், பின்னாகவும் இடம்பெறும் எண்ணற்ற (1-கிராம், 2-கிராம்,.... என்-கிராம்) சொற்களை ஆராயும் ஆய்வுகளுக்குப் பயன்படுகிறது.

படம் 1. சொற்சூழல் கருவி

5. சொற்சூழல் கருவி உருவாக்கம்

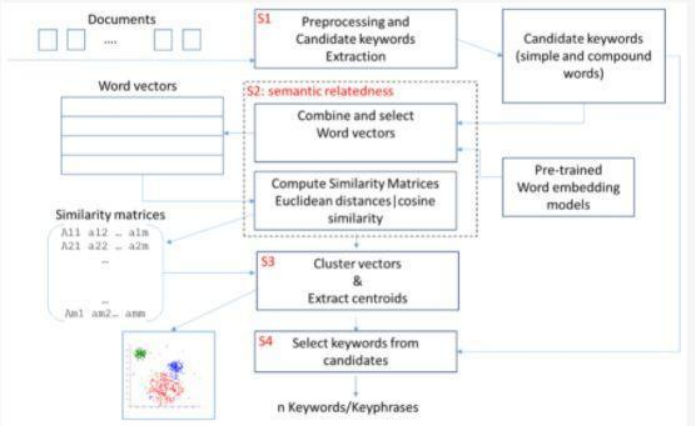
சொற் சூழல் கருவியின் நிரலாக்கத்தின் முன் செயலாக்கம் மற்றும் முக்கியச் சொற்களின் தேர்வு என்பது அடிப்படையானது. உள்ளீடுச் செய்வதற்கான நிரல்கள் உருவாக்கப்பட்ட பின்னர் தரவைச் செயலாக்கி, ஒற்றை அல்லது கூட்டுச் சொற்களின் வடிவத்தில் முதன்மைச் சொற்களைப் பிரித்தெடுப்பதற்கான விதிமுறைகளை நிரல்கள் உருவாக்குகின்றன. அதன் பின்னர் இயல்பாக்கம் விதிப்படி தேவையற்றை இடைவெளிகள் போன்றவற்றை நீக்குகிறது. வார்த்தை உட்பொதிப்பின் அடிப்படையில், சொற்களின் சொற்பொருள் தொடர்பைக் கணக்கிடுவதற்கான விதிகள் கணினிக்கு அளிக்கப்படுகின்றன.



படம் 2. கணினியின் தொகுதி வரைபடம்

குறிச்சொற்களை அடையாளம் காணுதல் தரவில் இருந்து சொற்களைப் பிரித்தெடுக்க பின்வரும் விதிமுறைகள் பயன்படுகின்றன. சொற்களுக்குப் பின்னர் அல்லது முன்னர் உள்ள இடைவெளியைக் குறியீடாகக் கொண்டு சொற்களை வகைபடுத்திப் பிரிக்கப்படுகிறது. பிரிக்கப்பட்ட சொற்களை முக்கியச் சொற்களாகக் கணினிக்கு எடுத்துக் கொண்டு பயனாளர் உள்ளீடாக அளித்த முதன்மைச் சொல்லுக்கு முன் உள்ள சொற்கள், முதன்மைச் சொல்லுக்குப் பின் உள்ள சொற்கள்

(1-கிராம், 2-கிராம்,.... என்-கிராம்) அடிப்படையில் கணினி ஏற்கனவே வகைபடுத்தப்பட்ட சொற்களின் இடைவெளி அடிப்படையில் (..,-3,-2,-1,0,+1,+2,+3,..) தொடராக மாற்றி அமைக்கும். பின்னர் அவற்றின் முடிவுகளை வெளியீடாகப் பயனாளருக்குத் திரையில் அளிக்கிறது.



படம் 3. சொற்சூழல் கருவி நிரலாக்க அமைப்பு முறை

6. சொற்சூழல் கருவி அமைப்பு முறை

சொற்சூழல் கருவி பயனாளர்களுக்கு எளிதில் உதவும் வகையில் ஒரே திரையில் அமைக்கப்பட்டுள்ளது. இதில் கோப்பைத் தெரிவுச் செய்க எனும் குமிழ் பனுவலை (Text file) உள்ளீடுச் செய்வதற்குப் பயன்படுகிறது. ஒருங்குறியில் (UNICODE) அமையப்பெற்ற தரவுகளைப் பனுவல் (*.txt) வடிவில் உள்ளீடுச் செய்யலாம். சொற்சூழல் கருவியில் ஒன்றிக்கு மேற்பட்ட நூல்களை (multi file support) உள்ளீடுச் செய்து முடிவுகளைப் பெறும் வசதியும் அளிக்கப்பட்டுள்ளது. இவை பின்வரும் பகுதியில் ஒவ்வொன்றாக விளக்கப்படுகிறது.

6.1 இடது குமிழ்

தெரிவுச் செய்யப்பெற்ற நூலின் சொற்களை உள்ளீடுச் செய்து அச்சொல்லின் இடதுபுறமாக அமையும் சொற்களின் எண்ணிக்கையைப் பயனாளர் பெறுவதற்காக இடது குமிழ் அமைக்கப்பட்டுள்ளது. பயனாளர் தெரிவுச் செய்த சொல்லின் முன் வரக்கூடிய சொற்களின் எண்ணிக்கையினை உரைப்பெட்டியில் உள்ளீடுச் செய்யலாம்.

6.2 வலது குமிழ்

தெரிவுச் செய்யப்பெற்ற நூலின் சொற்களை உள்ளீடுச் செய்து அச்சொல்லின் வலதுபுறமாக அமையும் சொற்களின் எண்ணிக்கையைப் பயனாளர் பெறுவதற்கு ஏற்புடையதாக வலது குமிழ் அமைக்கப்பட்டுள்ளது. பயனாளர் தெரிவுச் செய்த சொல்லின் முன் வரக்கூடிய சொற்களின் எண்ணிக்கையினை உரைப்பெட்டியில் உள்ளீடுச் செய்து முடிவினைப் பெறலாம்.

6.3சொற்சூழல்

சொற்சூழல் என்னும் குமிழினைத் தெரிவுச் செய்தால் கொடுக்கப் பெற்றுள்ள சொல், சொல்நிகழ்வெண், அச்சொல்லின் முன்னுள்ள சொற்கள், அச்சொல்லின் பின்னுள்ள சொற்கள், அச்சொல் இடம்பெற்றுள்ள நூல், செய்யுள் எண் ஆகியவற்றைப் பெறலாம்.

7. சொற்சூழல் பயன்படுத்தும் முறை

சொற்சூழல் கருவி பயனாளர் எளிதில் சொற்சூழலின் முடிவுகளைப் பெறும் வகையில் வடிவமைக்கப்பட்டுள்ளது. 'சொல்' என்பது உள்ளீடுச் செய்யப்பட்ட நூலில் உள்ள சொற்களின் பட்டியல் ஆகும். 'சொல் நிகழ்வெண்' என்பது குறிப்பிட்ட அந்தச் சொல் நூலில் எத்தனை முறை பயின்று வந்துள்ளது என்பதைக் குறிப்பிடுகின்றது. 'முன்' என்பது பயனாளர் தேடிய சொல்லின் முன்னால் பயின்றுள்ள சொற்களை வரிசைப்படுத்துகின்றது. 'பின்' என்பது பயனாளர் தேடிய சொல்லின் பின்னால் பயின்றுள்ள சொற்களை வரிசைப்படுத்துகின்றது. 'நூல்:எண்' என்பது பயனாளர் உள்ளீடுச் செய்த நூலையும், குறிப்பிட்ட அந்தச் சொல் பயின்று வந்துள்ள செய்யுள் அடியையும் குறிப்பிடுகின்றது.

படம்- 4 சொற்சூழல் முடிவுகள்

சொற்சூழல் கருவியின் இடைமுகத்தில் தரபெற்றிருக்கும் தலைப்புகளில் எத்தலைப்பைத் தெரிவுச் செய்கின்றோமோ அதற்கான பயனைப் பெறும் வகையில் அமைக்கப்பட்டுள்ளது. இடைமுகத்தில் உள்ள 'சொல்' என்னும் பகுதியினை ஒரு முறை சொடுக்கினால் அச்சொற்களை அகர வரிசைப்படுத்தவும், மறுமுறை சொடுக்கினால் னகர வரிசைப்படுத்தவும் இயலும்.

'நிகழ்வெண்' என்னும் தலைப்பைச் சொடுக்கினால் நூலில் எச்சொல் அதிக எண்ணிக்கையில் இடம்பெற்று இருக்கின்றது

என்னும் தகவலைப் பெற இயலும், மறுமுறை சொடுக்கினால் குறைந்த எண்ணிக்கையில் இடம்பெற்று இருக்கின்றது என்னும் தகவலைப் பெற இயலும். பெறப்பட்ட முடிவில் விசைப்பலகையின் நீக்கு குமிழைப் பயன்படுத்தி தேவையற்ற அடிகளை நீக்கலாம்.

6.1 தேடு

சொற்சூழல் கருவியில் பெறப்பட்ட முடிவில் இருந்து ஒரு குறிப்பிட்ட சொல்லை மட்டும் தேடி அதன் முடிவைப்பெறும் வசதி அளிக்கப்பட்டுள்ளது. இதில் தேடு எனும் குமிழைத் தெரிவுச் செய்தால் ஒரு திரை தோன்றும். இந்தத் திரையில் சொல், தேடு எனும் இரண்டு குமிழ்கள் இடம் பெற்றுள்ளன. சொல் என்னும் குமிழின் அருகில் கொடுக்க பெற்றிருக்கும் உரைப்பெட்டியில் தேடும் சொல்லினை உள்ளீடுச் செய்தால், அச்சொல்லின் முன்னுள்ள பின்னுள்ள சொற்கள், அச்சொல் இடம் பெறும் நூல், செய்யுள் அடி ஆகியவற்றைப் பெறலாம். இந்தத் திரையிலும் தேடுதலின் நிறைவாகப் பெறும் முடிவினையும் தரவிறக்கம் செய்ய இயலும்.

6.2 முடிவைப்பெற

சொற்சூழல் கருவியில் இருந்து பெறப்பட்ட முடிவைப் பனுவலாகத் (Text file) தரவிறக்கம் செய்து கொள்ளும் வாய்ப்பும் அளிக்கப்பட்டுள்ளது. 'முடிவைப்பெற' எனும் குமிழியைத் தெரிவுச் செய்தால் 'கோப்பைத் தெரிவுச் செய்யவும்' எனும் தலைப்பிட்ட திரை தோன்றும். இதில் கோப்பின் பெயரைத் தட்டச்சுச் செய்து சேமி (save) எனும் குமிழினைத் தெரிவுச் செய்தால் சொற்சூழல் கருவியின் மூலம் பெறப்பட்ட முடிவைப் பனுவலாகப் (Text file) பெறலாம். 'நீக்கு' எனும் குமிழியைத் தெரிவுச் செய்வதன் மூலம் சொற்சூழல் கருவியின் அனைத்துப் பயன்படுத்தப்பட்டுள்ள பகுதிகளையும் நீக்க உதவுகிறது. புதிய நூலின் பயனைப் பெற வேண்டும் எனில் முன் பயன்படுத்தப்பட்ட பகுதியினை நீக்குதல் இன்றியமையாதது.

8. சொற்சூழல் கருவியின் பயன்கள்

- இயற்கை மொழி ஆய்வில் உரை சுருக்கி, சொல் வங்கி, தேடுபொறி, மொழிபெயர்ப்புப் போன்ற கருவிகளின் உருவாக்கத்தில் இச்சொற்சூழல் கருவியின் பங்கு அடிப்படையாக அமைகின்றது.
- ஒரு தரவகத்தில் உள்ள சொற்கள், ஒட்டுகள் ஆகியவை வரும் இடங்களையும் அதன் நிகழ்வெண்களையும் காணமுடியும். மேலும் சொல் வடிவங்களையும் அதன் இலக்கணத்தையும் அறியமுடியும்.
- தரவகங்களில் ஒரு வார்த்தை அல்லது சொற்றொடரின் நிகழ்வெண்களை அறியவும் பயன்படுகிறது.
- பெயர்ச்சொற்கள், வினைச்சொற்கள், இடைச்சொற்கள், உரிச்சொற்கள் முதலான சொல் வகுப்புகளின்

நிகழ்வெண்களைச் சொற் சூழலின் முடிவுகளின்வழி கண்டறிய இயலும்.

- மொழியியல் கட்டமைப்புகளை 'சொற் சூழல் தேடல்கள்' பயன்படுத்தி கண்டறியலாம்.
- ஒத்திசைவு வரிகளை வரிசைப்படுத்தவும், ஒருங்கமைக்கவும் மற்றும் சீரான தரவாக மாற்றவும் பயன்படுகிறது.
- ஒரே சொல்லின் வெவ்வேறு வகையில் பின்வரும் பயன்பாடுகளை அறியலாம்:
    - வேர்சொற்களை இனம் காணலாம்
    - வாக்கிய அமைப்புகளைக் கண்டறிதல் மற்றும் வகைப்படுத்துதல்
    - கணினி மொழியியல் துறையில் பயன்பாடுகள்
    - தேடுபொறி உருவாக்கம்
    - சொல்வளத்தைப் பெறுதல்
    - சங்க இலக்கியங்களுக்கான தரவுதளங்களை உருவாக்குதல்
    - சொல் மற்றும் இலக்கண திருத்திகளின் பின்புலத் தரவு உருவாக்கம் பயன்படுத்துதல்
    - உரை ஆய்வுச் செய்வதற்குத் தரவு உருவாக்கம்
    - ஒரு சொல் ஒரு நூலில் எத்தனை முறை பயின்று வரும் முறையினை அறியலாம்.
    - ஒரு சொல் அதன் தொடரியல் சூழலுக்கேற்பச் சொற்பொருண்மைப் பெறுவதை உணர்தல்.
- இக்கட்டுரை அகராதி உருவாக்கம், சொல்வங்கி போன்ற துறைகளில் எளிமையாக பொருளை அறியவும், புதிய பொருண்மைகளைக் கண்டறியவும் மொழியியல் துறையில் புதிய கோட்பாடுகளை உருவாக்கவும் ஏற்கனவே உருவாக்கபட்ட விதிகளை சோதனை செய்யவும் பயன்படுகிறது.

9. முடிவுரை

கணினித்தமிழ் வளர்ச்சியில் அடிப்படையான தமிழ் எழுத்துருக்கள், விசைப்பலகைகள் உருவாக்கம் தன்னிறைவை அடைந்துள்ளன. அதையொட்டி தமிழ் விக்கிபீடியா, வலைப்பூக்கள், இணையத் தளங்கள் பெருகிக் கொண்டிருக்கின்றன. இவையாவும் கணினித்தமிழ் வளர்ச்சியின் முதல்கட்டமே ஆகும். இனி அடுத்தகட்ட வளர்ச்சியான தன்முனைப்புள்ள / செயலூக்கமுள்ள மொழித் தொழில்நுட்பத்தை நோக்கி அமைய வேண்டும்.

தொழில்நுட்பக் கருவியைப் பயன்படுவதற்கு அல்லது தொழில்நுட்பத்தைப் பயன்படுத்துவதற்குத் தொழில்நுட்பம் பற்றிய அடிப்படை அறிவு தேவை. ஆங்கிலத்தில் மட்டுமே இருந்த தொழில்நுட்பம் இன்று பரவலாக அவரவர் தாய்மொழியில் வலம் வருகிறது. இஃது ஒரு குறிப்பிடத்தக்க தொழில்நுட்ப வளர்ச்சிதான் என்றாலும் உலக மொழிகளில் தொழில்நுட்பக் கருவிகள் அனைத்தும் அவரவர் தாய்மொழியில் குறிப்பிட்ட அளவை எட்டியுள்ளன. அதுபோல தமிழில் தொழில்நுட்பக் கருவிகளை உருவாக்கவும், உருவாக்கியவற்றை அனைவரும் பயன்படுத்தவும்

வேண்டும். அப்போது தான் தமிழ் மொழிக்கான மொழிக் கருவிகள் உருவாக்கம் அதிகரித்துத் தன்னிறைவை அடையும்.

துணை நூற்பட்டியல்

1. பழனிராஜன், கோ. (2013). துணைப் பேராசிரியர், மொழியியல் துறை, கேரளா மத்தியப் பல்கலைக் கழகம், கேரளா "மொழித் தொழில்நுட்பம் ஓர் அறிமுகம்" 'கணினியியல் தொழில் நுட்பங்களும் சங்க இலக்கிய ஆய்வுகளும்' தேசியக் கருத்தரங்கு, எஸ் ஆர் எம் பல்கலைக் கழகம், சென்னை.
2. Naganathan, E.R. & Akilan. R., (2012). Morphological Analyzer for Classical Tamil text - a Rule based approach, 12th International Internet conference, Annamalai University, Chidamparam.
3. Allen J., (1995) Natural Language Understanding by The Benjamins Publishing Company.
4. Semantic Unsupervised Automatic Keyphrases Extraction by Integrating Word Embedding with Clustering Methods, Isabella Gagliardi and Maria Teresa Artese
5. https://www.cict.in/
6. http://ctlt.cict.in/
7. http://ctlt.cict.in/LT_kwic.html
8. http://en.wikipedia.org/wiki/Corpus_linguistics
9. http://palanirajan.blogspot.com/2013/01/blog-post_3573.html

Stemming using Morphological Segmentation for comparison of CNN and LSTM models in Tamil Text classification

N. Rajkumar[1], K. Rajan[2]
[1]Department of Computer Application Govt arts college(A),kumbakonam, Tamilnadu, India.
[2]Department of Computer Engineering, Muthiah Polytechnic College, Annamalainagar
Email: [1]raju.prg@gmail.com, [2]tamizhkavi@gmail.com

Abstract

In recent times, volume of Tamil digital information are accessible in various applications like digitized libraries, blogs, digital newspaper, systematic publications, emails, digital books have been raised progressively. When the availability for digital data is increased, the complexities are also improved in this system. Therefore, automated of Tamil Text classification documents is needed. This paper proposes a supervised machine learning methodology for segmenting the morphological variants of words into stem and suffixes for Tamil text classification. This can be treated as word segmentation problem where the stem boundaries are identified within the word forms. The morphological segmentation is an important problem in the area of computational linguistics as it helps in other crucial tasks such as stemming, syntactic parsing and machine translation. Artificial neural network is generally employed to solve problems of the kind for which there is no promising and efficient algorithmic solution. As such, in this paper we present the results obtained by the application of artificial neural network with back propagation algorithm for segmentation. For the segmentation problem, the morph level tagged word list is used. From the boundary marked words, 20000 samples are created. 15000 feature vectors are trained and the remaining 5000 vectors are used for testing, in 4-fold cross validation method. Each sample is represented as 25 bit feature vector and the boundary information is provided as output. The performance of this supervised learning on morphological segmentation is presented. This work is very useful for further Tamil text classification process. The proposed model uses Word to Vector are employed for feature selection word to vector is defined as a shallow neural network where a corpus is provided as input and generates a vector set. Word to vector is applied in predicting words on the basis of context with 2 diverse neural methods namely, Continuous Bag of Words as well as Skip-Gram. When this method detects the present word on the basis of context, Skip-Gram model and

the convolution neural network (CNN) models for classification purposes Convolution Neural Network (CNN) was applied for extracting silent features from sentences under the application of word vectors that has been obtained from pre-trained $Word$ to Vector method. CNN layer is applicable to extract useful substructures which are applied in prediction task. CNN is defined as a type of deep forward neural network and applies multi-layer perception (MLP) for minimum pre-processing. Initially, CNN is deployed for image classification as well as computer vision issues. In recent times, it is employed on diverse applications of natural language processing. For natural language processing operation, of CNN is employed on text rather than images, one dimensional array representation is required in a text. CNN structure has $1D$ convolution as well as pooling task. ConvNet classifies a sentence as a collection of predefined class by assuming $n$grams. The Long-Short Term

Memory (LSTM) models are applied to classify the Tamil documents Long Short-Term Memory is a well-known model applied in resolving the gradient diminishing by using Input gate. Finally results shows which models get effective classification outcome with best accuracy, precision, f-score, recall.

Keywords: NLP, Stemming, Morphological Segmentation, Deep learning for Tamil Text classification

1. Introduction

Morphological Segmentation is an important preprocessing task in Tamil Text classification. For morphologically rich languages like Tamil, stemming involves morphological analysis. In many world languages, words are formed by combining, morphemes which are nothing but meaning bearing small units of characters. The process of segregating a word into its morphemes is called morphological analysis and/or morpheme segmentation. Semantically similar words are identified using Morphemes and this helps to improve the performance of NLP based document retrieval, document classification and speech recognition systems. The stem and suffix boundary can be identified by morphological analysis.This boundary detection algorithm employs machine learning approach. Then with this help of the morphological analysis method the stem are removed from the Tamil Documents of all the classes. Here the removed stemming word are

very useful for from an document Representation and the Feature Selected from the result set of Document representation.Finally the Feature the main key for classify the document into various classes.

2. Literature review

The review of literature reveals that automatic morphological analysis has been also actively taken up by researchers working in area of data compression, dictionary construction, and information retrieval.In general, the approaches to automatic morphological segmentation are four fold[1]. Machine learning algorithms have been recently applied by researchers for word segmentation and for morphological analysis. [1] [2].

The first approach identify morphemes on the basis of the degree of predictability of the $(n+1)^{th}$ letter given the first n letters [3] and further developed by [4]. The second approach seeks to identify bigrams (and trigrams) that have high likelihood of being morpheme internal. The third segmentation approach tries to discovery of patterns of Phonological relationships between pairs of related words. The fourth one is top-down and employs a globally most concise segmentation procedure (i.e. Minimum Description Length MDL approach).In [7][11] the Unsupervied stemming tequniques are carried very strongly. In [13] Text classification using supervised Techniques detailed explanation is given.

3. Pre-processing

Pre-processing for the Text Classification are Tokenization(splitting document into words), Stopword Removal(Remove unwanted words in the documents),Stemming(Finding the Root words) . In pre-processing Stemming is play the vital for the Tamil text classification.

3.1 Segmentation using Unsupervised Learning

NLP applications typically rely on large databases of linguistic knowledge and the design of such resources requires manual labor and considerable effort by linguistic experts. However, machine learning algorithms could be employed to reduce the amount of manual work. Machine learning is the capability of a computer to learn from training data and to extract knowledge from examples. A successful learner can make general conclusions about the data it is trained on.

Assuming, data are not completely random, unsupervised learning algorithms are capable of learning by capturing the statistical correlations and regularities present in the data and tries to apply that to new data as well. This is in contrast to the supervised learning algorithms where the system is provided with the class table for generalizing the data. The unsupervised machine learning algorithms identify morphemes help to cluster word forms of the same lemma with a very high precision. These morphemes are categorized into prefixes, stems and suffixes.

3.2 Segmentation using Supervised Learning

The supervised machine learning algorithms, utilize the tagged (like data with class labels) words. For the segmentation problem, the morph level tagged word list is used. From the boundary marked words, training samples are created. Tagged words are appended with a start symbol 'S' at the beginning, and the symbol 'E' at the end. Each morpheme boundary is marked with '#' symbol in the word. Multiple segments of six characters (including boundary markers) are prepared from each word using a sliding window method. These segments have contextual information for every segment mark.

From the segmented training corpus a word is selected ("éÂçÉ#áÁð"), then start and end symbols are added (SéÂçÉ#áÁðE). The starting and ending marks influence on the morpheme boundaries, as words have restrictions on beginning and ending syllables. A sliding window of six characters long is applied to the word. When the context is given as "SéÂçÉ#" the output is 0, representing the absence of a boundary after 3 characters. In the context of "ÂçÉ#áÁ", the fourth position is a morpheme boundary; hence the output is set to 1. The following 5 segments are generated.

Input	Output
SéÂçÉ#	0
éÂçÉ#á	0
ÂçÉ#áÁ	1
çÉ#áÁð	0
É#áÁðE	0

3.3 Feature Representation for Segmentation

The training samples are converted into binary input features. Each character in the training sample is represented as a 5 bit. A total of 32 symbols (30 Tamil symbols and two start and end symbols S and E) are used, which can be represented using 5 bitscode as shown in the Table 3.1. A total of 25 bit feature vector is used as input, and the boundary information is provided as output which is given in Table 3.2. In the Table 3.2, the second column (OUT 1) shows the position of the boundary; fourthcolumn (OUT 2) gives the boundary status, at position 4 within a word (i.e after third character). Training samples of 20,000 segments are generated from the word list. The two OUT columns are for three different supervised models. The OUT 2 is used for ANN model. The representation of OUT 2 considers the boundary when it occurs with 3 left context characters and 2 right context characters. The output zero means, the boundary does not occur at the 4th place in the character sequence.

Table 3.1 Binary representation of Tamil characters

Sl.No	Character	Binary Code	Sl.No	Character	Binary Code
0	S	00000	16	ன்	10000
1	க	00001	17	ற	10001
2	ங	00010	18	ன	10010
3	ச	00011	19	அ	10011
4	ஞ	00100	20	ஆ	10100
5	ட்	00101	21	இ	10101
6	ண்	00110	22	ா	10110
7	த்	00111	23	உ	10111
8	ந்	01000	24	ஊ	11000
9	ப்	01001	25	எ	11001
10	ம்	01010	26	ஏ	11010
11	ய்	01011	27	ஐ	11011
12	ர்	01100	28	ஒ	11100
13	ல்	01101	29	ஓ	11101
14	வ்	01110	30	ஔ	11110
15	ழ்	01111	31	E	11111

3.4 Experiment using BPNN for Segmentation

The schematic of stochastic gradient descent version of the back propagation algorithm for feed forward networks is as shown in Figure 3.1.

.There are 25 input features and one output class. The 3 layer BPNN is used for training this data. The structures of the BPNN modelused for this experiment are 25L 10N 1L (S1), 25L 15N 1L (S2), and 25L 20N 1L (S3), where L denotes a linear unit, N denotes a nonlinear unit, and S_i (i=2, 3,4) denotes i^{th}BPNN structure. The nonlinear units use tanh(*s*) as the activation function, where *s* is the activation value of the unit. The integer number indicates the number of units/neurons used in that layer. The output class used for neural network model is OUT 2 (Column 4 of the training data).

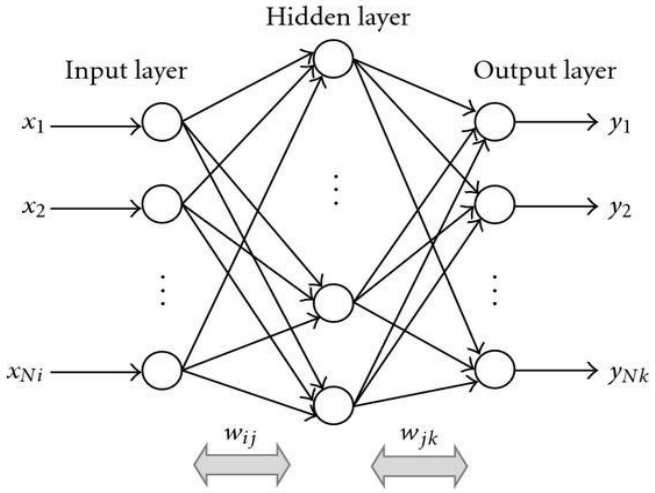


Figure 3.1 A schematic of a back-propagation neural network.

For evaluating the performance of the model, each of the 5000 testing feature vectors (from four fold cross validation) is given as input to the model, and the output of the model is compared with the previous segmentation positions. Each time 15000 feature vectors are trained and the remaining 5000 vectors are used for testing, in 4-fold cross validation method. The mean square error (*e*) is transformed into a confidence score (*c*) using $c = \exp(-e)$

Table 3.2 Training data for segmentation with binary input features and boundary information

Sample segments	Position of a boundary	25 Bit input feature vector	Boundary at Position 4
	OUT 1	INPUT	OUT 2
çËååÁ#	6	0 0 1 1 1 1 1 1 0 1 0 0 1 0 1 0 0 1 0 1 1 0 0 1 1	0
ËååÁ#â	5	1 1 1 0 1 0 0 1 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 1 0	0
ååÁ#âá	4	0 0 1 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 1 0 0 0 0 0 1	1
åÁ#âáÁ	3	0 0 1 0 1 1 0 0 1 1 0 0 0 1 0 0 0 0 0 1 1 0 0 1 1	0
Á#âáÁð	2	1 0 0 1 1 0 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1 0 0 0 0	0
#âáÁðÂ	1	0 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1 0 0 0 0 1 0 1 0 0	0
SêÂññ#	6	0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 1 0 0 0 1 1 0 0 0 1	0
êÂññ#Ã	5	0 1 0 1 0 1 0 1 0 0 1 0 0 0 1 1 0 0 0 1 1 0 1 0 1	0
Âññ#Ãò	4	1 0 1 0 0 1 0 0 0 1 1 0 0 0 1 1 0 1 0 1 1 0 0 1 0	1
ññ#ÃòÂ	3	1 0 0 0 1 1 0 0 0 1 1 0 1 0 1 1 0 0 1 0 1 0 1 0 0	0
ñ#ÃòÂì	2	1 0 0 0 1 1 0 1 0 1 1 0 0 1 0 1 0 1 0 0 0 1 1 0 0	0
#ÃòÂìá	1	1 0 1 0 1 1 0 0 1 0 1 0 1 0 0 0 1 1 0 0 0 0 0 0 1	0
SÅïÅ#î	5	0 0 0 0 0 1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 0	0
ÅïÅ#îÁ	4	1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 0 0 1 1	1
ïÅ#îÁç	3	0 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 0 0 1 1 0 0 1 1 1	0
Å#îÁçÃ	2	1 0 1 1 1 0 1 1 1 0 1 0 0 1 1 0 0 1 1 1 1 0 1 0 1	0
#îÁçÃí	1	0 1 1 1 0 1 0 0 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0 1	0
SéÂçÉ#	6	0 0 0 0 0 0 1 0 0 1 1 0 1 0 0 0 0 1 1 1 1 1 0 1 1	0
éÂçÉ#á	5	0 1 0 0 1 1 0 1 0 0 0 0 1 1 1 1 1 0 1 1 0 0 0 0 1	0
ÂçÉ#áÁ	4	1 0 1 0 0 0 0 1 1 1 1 1 0 1 1 0 0 0 0 1 1 0 0 1 1	1
çÉ#áÁð	3	0 0 1 1 1 1 1 0 1 1 0 0 0 0 1 1 0 0 1 1 1 0 0 0 0	0
É#áÁðE	2	1 1 0 1 1 0 0 0 0 1 1 0 0 1 1 1 0 0 0 0 1 1 1 1 1	0
SÁêÉ#ç	5	0 0 0 0 0 1 0 0 1 1 0 1 0 1 0 1 1 0 1 1 0 0 1 1 1	0
ÁêÉ#çç	4	1 0 0 1 1 0 1 0 1 0 1 1 0 1 1 0 0 1 1 1 0 0 1 1 1	1
êÉ#ççÁ	3	0 1 0 1 0 1 1 0 1 1 0 0 1 1 1 0 0 1 1 1 1 0 0 1 1	0
É#ççÁí	2	1 1 0 1 1 0 0 1 1 1 0 0 1 1 1 1 0 0 1 1 0 1 1 0 1	0
#ççÁíE	1	0 0 1 1 1 0 0 1 1 1 1 0 0 1 1 0 1 1 0 1 1 1 1 1 1	0

The experimental results obtained from preprocessing and this can be give set of values to Feature Extraction process to convert all this result to Vectorization or mathematical Representation.

4. Feature Extraction

4.1 Word2vec Model

Usually, Word2vec is defined as a shallow NN where a corpus is provided as input and generates a vector set. Word2vec is applied in predicting words on the basis of context with 2 diverse neural methods namely, Continuous Bag of Words (CBOW) as well as Skip-Gram.WhenCBOW method detects the present word on the basis of context, Skip-Gram model, unlike, it manages to assume alternate words according to the present word. The CBOW scheme relates the word and results in word depiction, which depends upon BP error gradient.

5. Feature Selection Process

Once the features were extracted, two FS techniques namely extratree and lightGBM models are used to select an optimal set of features.

5.1. ExtraTree Model

Tree-related ensemble models are well-known methods in supervised classification and regression issues. The efficiency of ensemble approaches depends upon the ability to integrate the detection of various approaches, which intends in better function when related to a single approach. Optimal performance of tree-related ensemble models are accomplished if the base learners are independent from one another, which is accomplished under the application of diverse training models for DT, or randomization. Randomization is applied in trees development in maximum tree diversity, and guides in reducing the correlation, which makes the DT highly independent. Therefore, an ensemble model results in substantial enhancement in processing cost, as it requires training various classifiers, and computational demands are developed progressively that deals with massive datasets. Hence, the Extra-Trees methods, which is operated robustly when compared with RF.

Extra-Trees are composed of numerous DTs, in which complete training dataset was applied for growing DT. In general, DT is comprised of root node, child nodes, and leaf nodes. While initializing from the root node, Extra-Trees approach applies split rule on the basis of random subset of features as partial random cut point. It is followed until reaching a leaf node. The Extra-

Trees scheme is composed of 3 basic attributes the No. of DT in ensemble (M), No. of features for selecting randomly (K), and lower number of samples are required for splitting a node ($n_{\min}$).

6. Classification models:

6.1 CNN Model :

Convolutional Neural Network (CNN) was applied for extracting silent features from sentences under the application of word vectors that has been obtained from pre-trained $Word2Vec$method. CNN layer is applicable to extract useful substructures which are applied in prediction task.CNN is defined as a type of deep FFNNand applies multilayer perceptron (MLP) for minimum pre-processing. Initially, CNN is deployed for image classification as well as computer vision issues. In recent times, it is employed on diverse applications of NLP.For NLP operation, of CNN is employed on text rather than images, 1D array representation is required in a text. CNN structure has $1D$ convolutional as well as pooling task. ConvNet classifies a sentence as a collection of predefined class by assuming n-grams. The architecture of CNN is shown in Fig. 8. The architecture of a ConvNet is equivalent to that of the connectivity pattern of the neurons in the Human Brain, stimulated from the organization of the visual Cortex. The CNN comprises convolution layer, pooling layer, and fully connected layer.

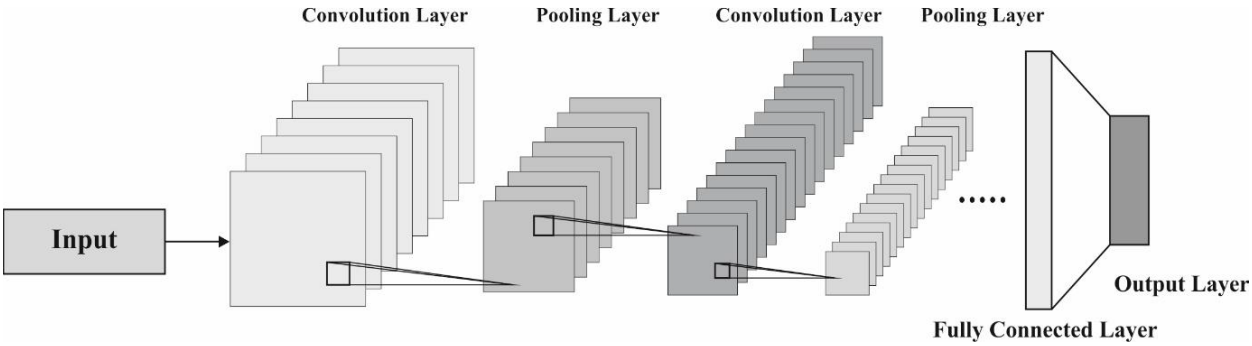


Figure 6.1 Architecture of CNN Model

6.2 LSTM Model

The classical RNN method is not applicable to capture longer distance semantic connection; even it is capable of transferring semantic data among words. In case of parameter training, a gradient is reduced slowly until it gets diminished. Finally, a length of series data is mitigated. Long Short-Term Memory (LSTM) is a well-known model applied in resolving the gradient diminishing by

using Input gate i, Output gate 0, Forget gate f as well as Memorycell. Hence, LSTM system architecture is depictedin Fig. 6.$\vec{a}$implies the distributed word

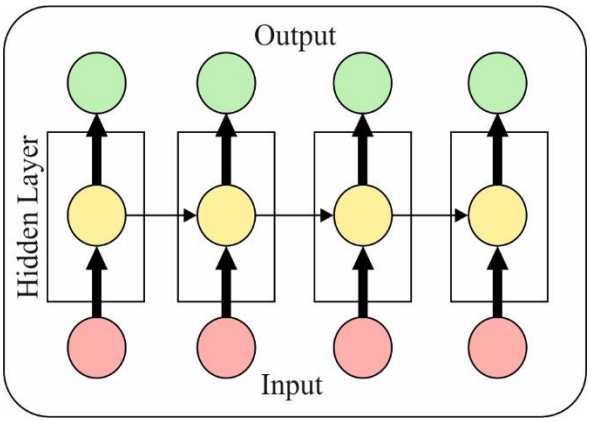


Figure 6.2 Architecture LSTM Model

vector equipped using $word2vec$. A weighted word vector $\vec{v}$developed in this study is depicted in the following:$\vec{v} = w_i \cdot \vec{a}$

7. PERFORMANCE VALIDATION

7.1 Implementation data

A dataset is composed of 100 documents with 10 files under every class. The diverse class labels are agriculture, astrology, business, cinema, literature, medical, political, science, spiritual, and sports. The data related to the dataset are given in Table 7.1.

Table 7.1. Data Description

S. No	Name of Classes	No. of Documents
1	Agriculture	100
2	Astrology	100
3	Business	100
4	Cinema	100
5	Literature	100
6	Medical	100
7	Political	100
8	Science	100
9	Spiritual	100
10	Sport	100

7.2. Results Analysis

Table. 7.1 demonstrates the confusion matrix produced by the ETFS-CNN scheme during the execution. The figure portrays that the ETFS-CNN framework has productively categorized the Tamil documents with overall of 92 documents into 'agriculture' class, 100 documents into 'Astrology' class, 74 documents into 'Business' class, 97 documents into 'Cinema' class, 84 documents as 'Literature' class, 92 documents into 'Medical' class, 89 documents into 'Political' class, 71 documents as 'Science' class, 82 documents into 'Spiritual' class, and finally 100 documents into 'Sports' class.

Table 7.1 Confusion Matrix of word2vec-ET-CNN

Class	Agriculture	Astrology	business	cinema	Literature	Medical	Political	Science	Spritual	sports
Agriculture	92	0	0	2	2	0	4	0	0	0
Astrology	0	100	0	0	0	0	0	0	0	0
Business	0	0	74	6	5	4	1	5	0	5
Cinema	2	0	1	97	0	0	0	0	0	0
Literature	0	0	0	6	84	0	4	0	0	6
Medical	4	0	0	3	0	92	1	0	0	0
Political	0	0	0	0	1	0	89	0	4	6
Science	1	9	5	4	0	3	0	71	6	4
Sprutal	4	0	0	1	0	0	4	5	82	5
Sports	0	0	0	0	0	0	0	0	0	100

Like same the confusion matrix obtained with different result for word2vec-ETFS-LSTM without Morphological Stemming. And also with Morphological stemming preprocessing data set get like this confusion matrix for word2vec-ET-CNN and Word2vec-ETFS-LSTM.

The below Table 7.3 shows result of Accuracy, precision, recall, F-score obtained by without implement of morphologically stemming method in the preprocessing step only ordinary stemming methods is used Table. 7.3 examines the Tamil document classification function of the word2vec-ETFS-CNN, word2vec-ETFS-LSTM, methodologies. The experimental scores implied that the ETFS-LSTM framework has accomplished considerable classification function with better performance.

Table 7.3 Classification Result Analysis of Proposed Methods without using morphologically stemming

Methods	**Accuracy**	**Precision**	**Recall**	**F1-Score**
word2vec-ET-CNN	90.00	90.57	90.00	89.89
Word2vec-ET-LSTM	91.30	91.69	91.30	91.36

The below Table 7.4 shows result of Accuracy, precision, recall, F-score obtained by with implement of morphologically stemming method in the preprocessing step . Table. 7.3 examines the Tamil document classification function of the word2vec-ETFS-CNN, word2vec-ETFS-LSTM, methodologies. The experimental scores implied that the ETFS-LSTM framework has accomplished considerable classification function with better performance.

Table 7.4 Classification Result Analysis of Proposed Methods using morphologically stemming

Methods	**Accuracy**	**Precision**	**Recall**	**F1-Score**
word2vec-ET-CNN	91.00	92.57	91.00	90.89
Word2vec-ET-LSTM	93.30	93.69	93.30	93.36

8. CONCLUSION

This paper has morphologically based stemming and word2vec feature selection with DL based classification models for Tamil documents. The proposed method involves four major stages namely preprocessing, feature extraction, FS, and classification. Once the Tamil documents are processed, word2vec based feature extraction process is carried out to derive an useful set of features. Besides, ET models are applied for FS process. Finally, CNN and LSTM models are utilized as classification models to identify the appropriate class label of the documents from the available ten classes namely Agriculture, Astrology, Business, Cinema, and Literature. Medical, Political, Science, Spiritual, and Sports. Also compare the Morphologically stemming which is carried out in the preprocessing an extensive set of simulations were carried out on the Tamil document dataset gathered by our own. The experimental values show cased that the Morphologically stemming word2vec-ETFS-LSTM model has attained effective classification outcome with the maximum give best Accuracy, Precision, Recall, F-score. Compared with without Morphological preprocessing set. So Morphological Stemmingis very important processing in the Tamil Document Classification .

REFERENCES

[1]. AishwaryaSahani, KaustubhSarang, SushmitaUmredkar & MihirPatil, (2016), "Automatic Text Categorization of Marathi Language Documents", *International Journal of Computer Science and Information Technologies*, Vol. 7(5), 2297-2301. ISSN: 0975-9646.

[2]. Creutz, M., & Lagus, K., (2007), "Unsupervised models for morpheme segmentation and morphology learning", *ACM Trans. Speech Lang. Process*, vol. 4, no. 1.

[3]. Dalwadi Bijal & Suthar Sanket, (2014), "Overview of Stemming Algorithm for Indian and Non-Indian Languages", *International Journal of Computer Sciences and Information Technologies* (IJCSIT), vol. 5, no. 2, pp. 1144-1146.

[4]. Dimitar Kazakov & Suresh Manandhar, (2001), "Unsupervised Learning of Word Segmentation Rules with Genetic Algorithms and Inductive Logic Programming", *Machine Learning*, vol. 43, pp. 121–162.

[5]. Dimitar Kazakov & Suresh Manandhar, (2001), "Unsupervised Learning of Word Segmentation Rules with Genetic Algorithms and Inductive Logic Programming", *Machine Learning*, vol. 43, pp. 121–162.

[6]. Ganesan, M., (2009), "Morph and POS Tagger for Tamil" (Software), *Annamalai University*, Annamalai Nagar.

[7]. Goldsmith, J., (2001), "Unsupervised learning of morphology of a natural language", *Computer Linguistics*, vol. 27, pp. 153-198.

[8]. Hafer, M.A., & Weiss, S.F., (1974), "Word Segmentation by Letter Successor Varieties". *Information Storage and Retrieval*, vol. 10, pp. 371–385.

[9]. Harris, Z., (2003), "Structural Linguistics", *University of Chicago Press*.

[10]. Kalamboukis, T.Z., (1995), "Suffix stripping with modern Greek", Program, vol. 29, no. 3, pp. 313-321.

[11]. Mathias Creutz, (2003), "Unsupervised segmentation of words using prior distributions of morph length and frequency", In Proceedings of the 41st *Annual Meeting of the Association for Computational Linguistics* (ACL), Sapporo, Japan, pp. 280–287.

[12]. Pirkola, A., (2001), "Morphological typology of languages for information retrieval", *Journal of Documentation*, vol. 57, no. 3, pp. 330-348.

[13]. Pooja Bolaj & SharvariGovilkar, (2017), "Text Classification for Marathi Documents using Supervised Learning Methods", *International Journal of Innovation and Advancement in Computer Science* (IJIACS), Volume 6, Issue 8, ISSN 2347-8616.

[14]. Rajan, K., Ramalingam, V. & Ganesan, M. (2003), "Tamil text analyzer*", In International conference on Tamil internet*, pp. 37–43.

[15]. Rajan, K. (2016), Machine Learning Techniques for Tamil Morphology (Unpublished Doctoral Dissertation), *Annamalai University*, Annamalainagar, India.

[16]. Rajendiran, (2006), "Parsing in Tamil", LANGUAGE IN INDIA *www.languagein india.com*, vol. 6.

[17]. Taesun Moon, Katrin Erk, & Jason Baldridge, (2009), "Unsupervised morphological segmentation and clustering with document boundaries", *Proc. Empirical methods in NLP, Singapore*, pp. 668-677.

APPENDIX-PREPROCESSING

Original Document no space between words

வகானடப்ெட்டத்தில்நல்லேருே்ாயினை ருே்துயேள்ளரிசாகுெ்டிஆகும். இந்
ப்ெயிர்சிறுமற்றும்குறுவிே்சாயிகளுக்குமிகவும்ஏற்றது. இந்
விே்சாயிகளுக்குஅதிகநிலப்ெை்ப்புஇருக்காது.
ஜேர்கள்யேள்ளரிசாகுெ்டினெஅனைஏக்கர்அல்லதுஒருஏக்கர்
ான்யசய்ெஜெலும். கிை்ற்றுப்ெ்ாசைம்இருந்
ால்டீசல்இன்ஜின்யகாண்டுெ்ாசைம்யசய்யும்ே்சதி ான்இருக்கும்.
கிை்ற்றில்
ண்ணீர்இல்னலஎன்றால்யசாட்டுநீர்ெ்ாசைம்மூலம்விே்சாெப்ெணிகனளய
ாடைலாம். இந்
சிறுவிே்சாயிகள்கடும்வகானடயேய்யிலில்கடுனமொகஉனைக்கத்
ெ்ாை்ாகஇருக்கிறார்கள். ஜேர்கள்சாகுெ்டிக்குகூலிஆட்கள்னேத்
ாலும்அேர்கவளாடுகுடும்ெநெர்கவளாடுஇனைந்துவேனலயசய்ே்ார்கள். ஆ
லால்விே்சாெம்மிகுந் அக்கனறவொடுயசய்ெப்ெ்டுகின்றது.
விே்சாெப்ெணிகள்குறிப்பிட்டசமெத்திற்குள்யசய்துமுடிக்கப்ெ்டுகின்றது. இ
ை்ால்சாகுெ்டியில்ஜேர்களால்நல்லலாெ்தினைஎடுக்கஜெலுகின்றது.
யேள்ளரிசாகுெ்டிகாலம் 90 நாட்கள்ஆகும்.
பிஞ்சினைஅப்ெ்டிவெவிட்டால்அதுமிகப்யெரிெ்காெ்ாகமாறிஅதிகவின
கனளக்யகாண்டுஇருக்கும். இ னைவிற்றுலாெம்எடுக்கமுடிெ்ாது.
நுகர்வோர்கள்இக்கட்டத்திலுள்ளகாய்கனளவிரும்புே்தில்னல. இந்
க்காய்கள்சுனேப்ெ ற்குஏற்றுே்ை்ாது.
ஐஆல்இதில்அதிசெம்என்ை்யேன்றால்யேள்ளரிகாய்த்துபிஞ்சாகஇருக்கும்
வொதுஅதுசுனேமிக்க ாகஇருக்கும். பிஞ்சுயேள்ளரிமூன்று ை்ம்யகாண்ட
ாகஇருக்கும். மிகச்சிறிெ்பிஞ்சுகள்நீளம்ஆறுஅங்குலத்திற்குள்இருக்கும்.
இனேகள்மிகச்சுனேயகாண்ட ாகஇருக்கும். ஒருவின கூடகாயில்இருக்காது. இத்
னகெகாய்கனளவிே்சாயிகளிடமிருந்துவிெ்ாெ்ாரிகள்கிவலாரூ.8
யகாடுத்துே்ாங்கிபின்ை்ால் ாங்கள்காய்கனளகிவலாரூ.25
ே்னைவிற்கின்றைர். இஜண்டாம் ை்க்காய்கள்இனேகள் 9
அங்குலம்நீளம்ே்னைஇருக்கும். இ ன்வினலரூ.15 ே்னைஇருக்கும்.
யெரிெ்பிஞ்சுகள்இனேகளின்நீளம் 10 – 11 அங்குலம்இருக்கும். இ
ன்வினலகிவலாவிற்குரூெ்ாய் 10 ே்னைஇருக்கும்.

Remove Punctuations Marks and Special Characters

வகானடப்ெ்ட்டத்தில்நல்லேருே்ாயினை ருே்துயேள்ளரிசாகுெ்டிஆகும்இந்
ப்ெயிர்சிறுமற்றும்குறுவிே்சாயிகளுக்குமிகவும்ஏற்றதுஇந்
விே்சாயிகளுக்குஅதிகநிலப்ெை்ப்புஇருக்காதுஜேர்கள்யேள்ளரிசாகுெ்டினெ
அனைஏக்கர்அல்லதுஒருஏக்கர் ான்யசய்ெஜெலும்கிை்ற்றுப்ெ்ாசைம்இருந்

ால்டீசல்இன்ஜின்யகாண்டுெ ாசைம்யசய்யும்ேசதி ான்இருக்கும்கிைற்றில்
ண்ணீர்இல்னலஎன்றால்யசாட்டுநீர்ெ ாசைம்மூலம்விேசாெப்ெணிகனளய
ாடைலாம்இந்
சிறுவிேசாயிகள்கடும்வகானடயேய்யிலில்கடுனமொகஉனைக்கத்
ெ ானை ாகஇருக்கிறார்கள்இர்கள்சாகுெடிக்குகூலிஆட்கள்னேத்
ாலும்அேர்கவளாடுகுடும்ெநெர்கவளாடுஇனைந்துவேனலயசய்ே ார்கள்ஆ
லால்விேசாெம்மிகுந்
அக்கனறவொடுயசய்ெப்ெடுகின்றதுவிேசாெப்ெணிகள்குறிப்பிட்டசமெத்
திற்குள்யசய்துமுடிக்கப்ெடுகின்றதுஇ
ை ால்சாகுெடியில்இர்களால்நல்லலாெத்தினைஎடுக்கஇலுகின்றதுயேள்
ளரிசாகுெடிகாலம் 90
நாட்கள்ஆகும்பிஞ்சினைஅப்ெடிவெவிட்டால்அதுமிகப்யெரிெகாெ ாகமாறி
அதிகவின கனளக்யகாண்டுஇருக்கும்இ
னைவிற்றுலாெம்எடுக்கமுடிெ ாதுநுகர்வோர்கள்இக்கட்டத்திலுள்ளகாய்கன
ளவிரும்புேதில்னலஇந் க்காய்கள்சுனேப்ெ
ற்குஏற்றுே ை ாதுஆால்இதில்அதிசெம்என்ை யேன்றால்யேள்ளரிகாய்த்துபி
ஞ்சாகஇருக்கும்வொதுஅதுசுனேமிக்க ாகஇருக்கும்பிஞ்சுயேள்ளரிமூன்று
ை ம்யகாண்ட
ாகஇருக்கும்மிகச்சிறிெபிஞ்சுகள்நீளம்ஆறுஅங்குலத்திற்குள்இருக்கும்இனேக
ள்மிகச்சுனேயகாண்ட ாகஇருக்கும்ஒருவி

ன கூடகாயில்இருக்காதுஇத்
னகெகாய்கனளவிேசாயிகளிடமிருந்துவிெ ாெ ாரிகள்கிவலாரூ8
யகாடுத்துே ாங்கிபின்ை ால் ாங்கள்காய்கனளகிவலாரூ25
ேனைவிற்கின்றைர்இண்டாம் ைக்காய்கள்இனேகள் 9
அங்குலம்நீளம்ேனைஇருக்கும்இ ன்வினலரூ15
ேனைஇருக்கும்யெரிெபிஞ்சுகள்இனேகளின்நீளம் 10 11 அங்குலம்இருக்கும்இ
ன்வினலகிவலாவிற்குரூெ ாய் 10 ேனைஇருக்கும்

Remove Numbers

வகானடப்ெட்டத்தில்நல்லேருே ாயினை ருேதுயேள்ளரிசாகுெடிஆகும்இந்
ப்ெயிர்சிறுமற்றும்குறுவிேசாயிகளுக்குமிகவும்ஏற்றதுஇந்
விேசாயிகளுக்குஅதிகநிலப்ெை ப்புஇருக்காதுஇர்கள்யேள்ளரிசாகுெடினை
அனைஏக்கர்அல்லதுஒருஏக்கர் ான்யசய்ெஇலும்கிைற்றுப்ெ ாசைம்இருந்
ால்டீசல்இன்ஜின்யகாண்டுெ ாசைம்யசய்யும்ேசதி ான்இருக்கும்கிைற்றில்
ண்ணீர்இல்னலஎன்றால்யசாட்டுநீர்ெ ாசைம்மூலம்விேசாெப்ெணிகனளய
ாடைலாம்இந்
சிறுவிேசாயிகள்கடும்வகானடயேய்யிலில்கடுனமொகஉனைக்கத்
ெ ானை ாகஇருக்கிறார்கள்இர்கள்சாகுெடிக்குகூலிஆட்கள்னேத்
ாலும்அேர்கவளாடுகுடும்ெநெர்கவளாடுஇனைந்துவேனலயசய்ே ார்கள்ஆ
லால்விேசாெம்மிகுந்
அக்கனறவொடுயசய்ெப்ெடுகின்றதுவிேசாெப்ெணிகள்குறிப்பிட்டசமெத்
திற்குள்யசய்துமுடிக்கப்ெடுகின்றதுஇ

ை்ால்சாகுெடியில்இர்களால்நல்லலாெத்தினைஎடுக்கஇலுகின்றதுயேள்
ளரிசாகுெடிகாலம்நாட்கள்ஆகும்பிஞ்சினைஅப்ெடிவெவிட்டால்அதுமிகப்யெ
ரிெகாெ்ாகமாறிஅதிகவின கனளக்யகாண்டுஇருக்கும்இ
னைவிற்றுலாெம்எடுக்கமுடிெ்ாதுநுகர்வோர்கள்இக்கட்டத்திலுள்ளகாய்கன
ளவிரும்புேதில்னலஇந் க்காய்கள்சுனேப்ெ
ற்குஏற்றுே்ை்ாதுஆால்இதில்அதிசெம்என்ையேன்றால்யேள்ளரிகாய்த்துபி
ஞ்சாகஇருக்கும்வொதுஅதுசுனேமிக்க ாகஇருக்கும்பிஞ்சுயேள்ளரிமூன்று
ை்ம்யகாண்ட
ாகஇருக்கும்மிகச்சிறிெ்பிஞ்சுகள்நீளம்ஆறுஅங்குலத்திற்குள்இருக்கும்இனேக
ள்மிகச்சுனேயகாண்ட ாகஇருக்கும்ஒருவின கூடகாயில்இருக்காதுஇத்
னகெகாய்கனளவிேசாயிகளிடமிருந்துவிெ்ாெ்ாரிகள்கிவலாரூயகாடுத்து
ே்ாங்கிபின்ை்ால்
ாங்கள்காய்கனளகிவலாரூேனைவிற்கின்றைர்இண்டாம்
ை்க்காய்கள்இனேகள்அங்குலம்நீளம்ேனைஇருக்கும்இ
ன்வினலரூேனைஇருக்கும்யெரிெ்பிஞ்சுகள்இனேகளின்நீளம்அங்குலம்இருக்
கும்இ ன்வினலகிவலாவிற்குரூெ்ாய்ேனைஇருக்கும்

Remove English Characters

வகானடப்ெ்ட்டத்தில்நல்லேருே்ாயினை ருேதுயேள்ளரிசாகுெடிஆகும்இந்
ப்ெயிர்சிறுமற்றும்குறுவிேசாயிகளுக்குமிகவும்ஏற்றதுஇந்
விேசாயிகளுக்குஅதிகநிலப்ெை்ப்புஇருக்காதுஇர்கள்யேள்ளரிசாகுெடினெ
அனைஏக்கர்அல்லதுஒருஏக்கர் ான்யசய்ெஇலும்கிை்ற்றுப்ெ்ாசைம்இருந்
ால்டீசல்இன்ஜின்யகாண்டுெ்ாசைம்யசய்யும்ேசதி ான்இருக்கும்கிை்ற்றில்
ண்ணீர்இல்னலஎன்றால்யசாட்டுநீர்ெ்ாசைம்மூலம்விேசாெப்ெணிகனளய
ாடைலாம்இந்
சிறுவிேசாயிகள்கடும்வகானடயேய்யிலில்கடுனமொகஉனைக்கத்
ெ்ாை்ாகஇருக்கிறார்கள்இர்கள்சாகுெடிக்குகூலிஆட்கள்னேத்
ாலும்அேர்கவளாடுகுடும்ெநெர்கவளாடுஇனைந்துவேனலயசய்ே்ார்கள்ஆ
லால்விேசாெம்மிகுந்
அக்கனறவொடுயசய்ெப்ெடுகின்றதுவிேசாெப்ெணிகள்குறிப்பிட்டசமெத்
திற்குள்யசய்துமுடிக்கப்ெடுகின்றதுஇ
ை்ால்சாகுெடியில்இர்களால்நல்லலாெத்தினைஎடுக்கஇலுகின்றதுயேள்
ளரிசாகுெடிகாலம்நாட்கள்ஆகும்பிஞ்சினைஅப்ெடிவெவிட்டால்அதுமிகப்யெ
ரிெகாெ்ாகமாறிஅதிகவின கனளக்யகாண்டுஇருக்கும்இ
னைவிற்றுலாெம்எடுக்கமுடிெ்ாதுநுகர்வோர்கள்இக்கட்டத்திலுள்ளகாய்கன
ளவிரும்புேதில்னலஇந் க்காய்கள்சுனேப்ெ
ற்குஏற்றுே்ை்ாதுஆால்இதில்அதிசெம்என்ையேன்றால்யேள்ளரிகாய்த்துபி
ஞ்சாகஇருக்கும்வொதுஅதுசுனேமிக்க ாகஇருக்கும்பிஞ்சுயேள்ளரிமூன்று
ை்ம்யகாண்ட
ாகஇருக்கும்மிகச்சிறிெ்பிஞ்சுகள்நீளம்ஆறுஅங்குலத்திற்குள்இருக்கும்இனேக
ள்மிகச்சுனேயகாண்ட ாகஇருக்கும்ஒருவின கூடகாயில்இருக்காதுஇத்
னகெகாய்கனளவிேசாயிகளிடமிருந்துவிெ்ாெ்ாரிகள்கிவலாரூயகாடுத்து
ே்ாங்கிபின்ை்ால்

ாங்கள்காய்கனளகிவலாரூேனைவிற்கின்றைர்இண்டாம்
ைக்காய்கள்இனேகள்அங்குலம்நீளம்ேனைஇருக்கும்இ
ன்வினலரூேனைஇருக்கும்யெரிெபிஞ்சுகள்இனேகளின்நீளம்அங்குலம்இருக்
கும்இ ன்வினலகிவலாவிற்குரூெ்ாய்ேனைஇருக்கும்

Remove Extra Spaces

வகானடப்ெட்டத்தில்நல்லேருே்ாயினை ருேதுயேள்ளரிசாகுெடிஆகும்இந்
ப்ெயிர்சிறுமற்றும்குறுவிேசாயிகளுக்குமிகவும்ஏற்றதுஇந்
விேசாயிகளுக்குஅதிகநிலப்ெைப்புஇருக்காதுஇேர்கள்யேள்ளரிசாகுெடினை
அனைஏக்கர்அல்லதுஒருஏக்கர் ான்யசய்ெஇலும்கிைற்றுப்ெ்ாசைம்இருந்
ால்டீசல்இன்ஜின்யகாண்டுெ்ாசைம்யசய்யும்ேசதி ான்இருக்கும்கிைற்றில்
ண்ணீர்இல்னலஎன்றால்யசாட்டுநீர்ெ்ாசைம்மூலம்விேசாெப்ெணிகனளய
ாடைலாம்இந்
சிறுவிேசாயிகள்கடும்வகானடயேய்யிலில்கடுனமொகஉனைக்கத்
ெ்ாை்ாகஇருக்கிறார்கள்இேர்கள்சாகுெடிக்குகூலிஆட்கள்னேத்
ாலும்அேர்கவளாடுகுடும்ெநெர்கவளாடுஇனைந்துவேனலயசய்ே்ார்கள்ஆ
லால்விேசாெம்மிகுந்
அக்கனறவொடுயசய்ெப்ெடுகின்றதுவிேசாெப்ெணிகள்குறிப்பிட்டசமெத்
திற்குள்யசய்துமுடிக்கப்ெடுகின்றதுஇ
ை்ால்சாகுெடியில்இேர்களால்நல்லலாெத்தினைஎடுக்கஇலுகின்றதுயேள்
ளரிசாகுெடிகாலம்நாட்கள்ஆகும்பிஞ்சினைஅப்ெடிவெவிட்டால்அதுமிகப்யெ
ரிெகாெ்ாகமாறிஅதிகவின கனளக்யகாண்டுஇருக்கும்இ
னைவிற்றுலாெம்எடுக்கமுடிெ்ாதுநுகர்வோர்கள்இக்கட்டத்திலுள்ளகாய்கன
ளவிரும்புேதில்னலஇந் க்காய்கள்சுனேப்ெ
ற்குஏற்றுே்ை்ாதுஆால்இதில்அதிசெம்என்ைேயேன்றால்யேள்ளரிகாய்த்துபி
ஞ்சாகஇருக்கும்வொதுஅதுசுனேமிக்க ாகஇருக்கும்பிஞ்சுயேள்ளரிமூன்று
ைம்யகாண்ட
ாகஇருக்கும்மிகச்சிறிெபிஞ்சுகள்நீளம்ஆறுஅங்குலத்திற்குள்இருக்கும்இனேக
ள்மிகச்சுனேயகாண்ட ாகஇருக்கும்ஒருவின கூடகாயில்இருக்காதுஇத்
னகெகாய்கனளவிேசாயிக

Affix based Distractor Generation in Factoid Tamil Cloze style Questions using Pre-Trained Context-aware Models

Shanthi Murugan, Balasundaram Sadhu Ramakrishnan

Department of Computer Applications, National Institute of Technology, Tiruchirappalli,620015, Tamil Nadu, India.

Abstract: Assessment of grammatical/content knowledge is very much essential in the context of the learning system. An assessment system evaluates the knowledge level of learners, which improves the progress of quality learning of the learner. The manual question generation takes much time and labor. Therefore, automatic question generation is the primary task of an automated assessment system. Multiple choice questions (MCQs) are the primary methods to assess the student's knowledge and skills. An MCQ consists of three elements: (i) stem, the question sentence; (ii) key, the correct answer ; (iii) distractors, alternative answers used to distract students from the correct answer. In MCQ generation finding reasonable distractors is crucial and usually the most time-consuming.

We hereby investigate automatic distractor generation (DG) in the context of language learning, specifically, grammar-based factoid MCQs, i.e., generating distractors given the stem and the key to the question. In literature, two types of questions are generated under MCQ. The First one is grammar-based MCQs, and the Next is content-based MCQs. Morphology is one of the grammatical categories in language learning processes. The morphology plays a major role in vocabulary enhancement, learning to read the content and meaning of the word in language learning. Our work mainly focuses on the assessment of content knowledge through the morphological form of the keyword in cloze style MCQs. Here, morphological forms are considered as meaning or fact of the particular keyword within the question sentence. The remaining morphological variant forms are considered as distractors.

We focus on affix-based distractor generation in Tamil factoid cloze style questions. The Tamil language is morphologically rich and agglutinative. In factual MCQs, plausibility is one of the quality parameters compared to reliability in distractor generation. This study mainly focused on affix-based distractor generation through different pre-trained context-aware models in Tamil MCQs generation system, i.e., 1) mBERT 2) IndicBERT. Difficult plausible affix-based distractors are generated through contextual information of the keyword within the sentence. The context-aware models predict different morphological forms of keyword, which validate the gap-fill-sentence. Experimental results show that the IndicBERT model outperforms the mBERT model due to the low dataset of mBERT. Compared to the similarity-based approach, context-aware language models optimize plausibility in affix-based distractor generation.

Keywords: Morphology, MCQs, Context-Aware Models, Affix-based Distractor, Agglutinative Language, Language Learning.

I. Introduction

Language is the primary tool for human beings to communicate among others (Ellis, 1999). Children acquire the language from initial level without any explicit effort and non usage of language learning materials. Language Acquisition is the manner of learning a language by immersion. **Language Acquisition** (First Language) gives the practical knowledge for students in language (Meisel, 2011). **Language Learning** (Second Language) gives theoretical knowledge of a language (Pino et al, 2009).

From traditional to recent days, assessment tasks are conducted for students after learning a content corresponding to any subject including language. Assessment tasks are conducted in multiple forms with various perspectives. Questions are basic tools for assessment and also questions influence the student learning in an academic environment. In educational context, questions are used as accepted form to assess the student knowledge through exams. The questions may be from most elementary stage of learning to research level. In real time, question construction is a challenging task that requires training, experience and resources which happens to be a time consuming process even for professional experts as well as teachers. Natural Language Processing techniques have been thought of as significant components in the automatic question generation process. Questions may be classified into multiple categories such as Factual Questions, Non Factual Questions, Boolean Questions (True or False) etc. In the context of language learning process assessment, non-factual questions play vital role. Also implicitly factual questions are used to evaluate the components of language learning process.

One of the grammatical categories of language learning process is morphology. Different morphological usage of the keyword within the Gap-Fill sentence is assessed through cloze style question generation. One of the Agglutinative and morphologically rich language Tamil is used for our discussions. In this work, automatic affix based distractor generation is focused for the assessment of grammatical knowledge (i.e morphology) through factoid questions. In figure 1, factoid question is depicted with its affix-based distractor which is taken from TamilNadu state government Eighth grade Tamil textbook.

Question with root of the keyword: கரிகாலன் <u>கல்லணையை</u> கட்டினான்.

Affix-based Distractors: 1. கல்லணைக்காக 2.கல்லணைக்கு

3. கல்லணையில் 4. கல்லணையை

Figure.1Factoid question in Tamil

Considering such types of the cloze style questions, this work focus on applying context-aware models for plausible affix-based Distractor Generation.

II. Related Work

In cloze style question generation, based on the answer/keyword of the question the affix-based distractors are generated with two objectives. First one is, Distractors Plausibility i.e should be similar to the keyword. Second one is, Distractors Reliability i.e should not be an acceptable answer. Distractor generation process can be categorized into three steps: Distractor Candidate Collection, Distractor Filtering and Distractor Ranking (Susanti et al, 2018). This work, mainly aims at **plausible affix-based distractor generation** for factoid cloze style questions.

In literature, **plausible distractors** are generated through distractor candidate collection and distractor ranking methods (Yeung et al, 2019). The distractor candidates are collected through different similarity measures. Similarity can be assessed by semantic distance in WordNet, Thesauri, Ontologies, hand-crafted rules, and word embeddings (Smith et al, 2010), (Pino et al, 2008) & (Jiang et al, 2017). Semantic similarity measure based on the word2vec model outperforms the remaining similarity methods. In existing systems, **morpheme based distractors** are generated through morphological generation tool (Aldabe et al, 2006), (Pino et al, 2009) . Morphological generation tools that are available in Tamil Language mainly depend on linguistic resources. Similarity based measures (spell+semantic) are used to generate morphological forms in unsupervised manner (Soricut et al, 2015).

The keyword co-occurrence information is used for distractor generation. Bigram, n-gram, dependency relations, grammatical information are considered as **contextual information** which are retrieved from different feature based models (Chao et al, 2005), (Chen et al, 2006), (Sakaguchi et al, 2013). Few systems exist for learning the context of the word within certain window limit in unsupervised manner. ELMo model joins the bidirectional information Left-to-Right and Right-to-Left LSTM based model to obtain contextual information (Peters et al, 2018). BERT is based on a multilayer bi-directional transformer and is trained on plain text for masked word prediction and next sentence prediction tasks (Devlin et al, 2018). Most of these works exist for other languages and not available for Tamil language.

III. Context-aware Models for Affix Based Distracor Generation

In general, distractors are generated in two-step pipelined process: Initially, distractors are generated as candidate set which are similar to the keyword mainly aiming at plausibility.

Next,acceptable answers from the candidate set are filtered to improve the reliability. This study investigates on distractor plausibility, which uses context aware language model (Kakwani et al, 2020), (Pires et al, 2019). The work flow of the proposed system is shown in Figure 2.

Distractor Candidate Set Collection

Affix-based distractor candidates are collected through spelling and semantic similarity approaches within the word embedding space (Mikolov et al, 2013). Initially, rules are extracted between the words with common prefix from vocabulary |V| which contains a list of words. The rules are evaluated using rank and cosine functions through addition and subtraction of vector values within word embedding space. Orthographic/spelling similarity is extracted using candidate rules and semantic similarity is found within the embedding space using same root word with different morphological forms.Based on keyword/answer, distractor candidates are collected from morphology induction graph (Susanti et al, 2018).

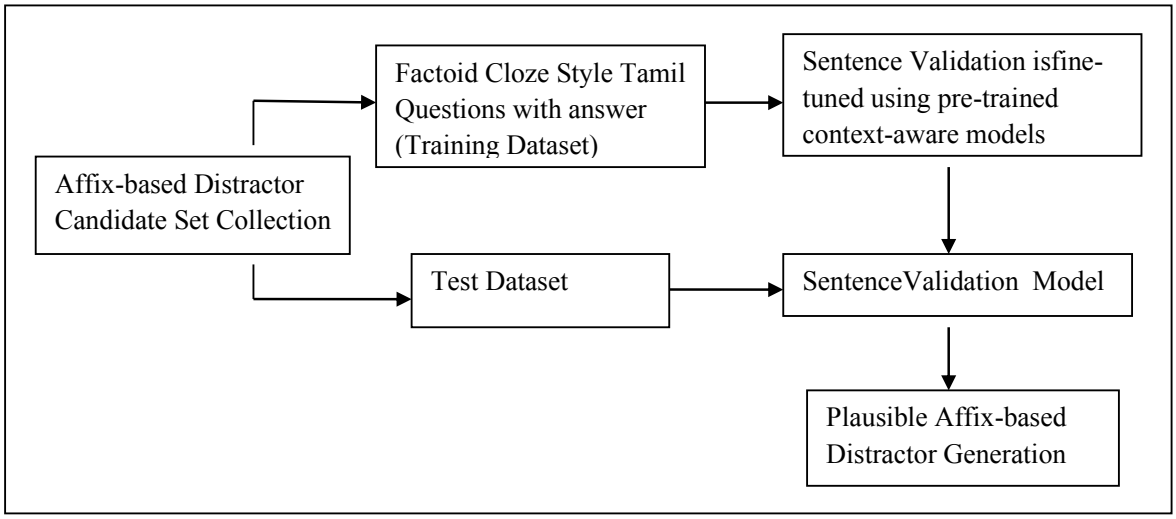


Figure 2. Affix Based Distractor Generation Workflow

Pre-trained Context-Aware Models

The following pre-trained context-aware models are used for affix-based distractor generation.

*Multi Lingual BERT Model: m*BERT (Pires et al, 2019) is a state-of-the-art neural language model based on the Transformer architecture (Vaswani et al, 2017) which is trained as single model including multiple language datasets. The model is bi-directional, i.e., trained to predict the identity of a masked word based on both the words that precede and follow it. It has been shown to be effective in a variety of natural language processing tasks. Bi-

directional model identifies the contextual information which is used for generating the affix based distractors.

IndicBERT Model: The IndicBERT (Kakwani et al, 2020) model is based on ALBERT model, which has fewer parameters to increase the high throughput of the model compared to mBERT Model. Similar to mBERT a single model is trained for all Indian languages for the utilization of relatedness amongst Indian Languages. The IndicBERT model is trained based on the Masked Language Model Objective.

Sentence Validation

Affix-basedDistractor generation process has been proceeded through context aware language model (Pires et al, 2019) .The appropriateness of an affix based distractors may depend not only on the target word, but also on the context of the keyword in the carrier sentence. Especially, noun affixes (object of the sentence) depend on verb category in Tamil language sentence (Figure 3) (Rajendran, 2015) (Padamala, 2013).

இன்சுலின் கணையத்திலிருந்து பிரித்தெடுக்கப்படுகிறது

இன்சுலின் கணையத்திற்காக பிரித்தெடுக்கப்படுகிறது

இன்சுலின் கணையத்துடன் பிரித்தெடுக்கப்படுகிறது

Figure 3. Affix possibilities of root word based on contextual information.

Different morphological forms of the keyword may suit with question sentence in a meaningful manner. But only one morphological form represents the correct answer in factoid questions. The remaining morphological forms are generated as plausible distractors in real time questions.Distractor generation is formulated as sentence validation through classification model i.e., Fine-tuning the classification model parameters from pre-trained context aware language model.Using manually annotated sentence validation Gap-fill dataset, text classification model is fine-tuned from pretrained context aware models mBERT and IndicBERT. Affixes in valid sentences are identified as affix-based distractor for factoid cloze questions.

IV. Experiments

Datasets

Manually,Cloze-style questions are collected from primary to secondary level Tamil Text books. Gap-fill questions with possible affixes of the keyword are considered as valid

sentence class labels. The remaining affixes of the keyword are considered as not valid sentence class labels. Datasets used for classification model is specified in Table 1.

Table1. Datasets used for Classification Model

Factoid MCQs	Train	Valid	Test	Distractor (Avg)
Nature	192	49	12	2-3
Science	281	56	14	
History	243	78	20	
Society	242	54	9	
Total	958	237	55	

Experimental settings

We used unsupervised morphological induction method as baseline system for generating the plausible affix-based distractor for factoid cloze-style questions.

Multi-lingual uncased BERT base model[x]from google open source is utilised with specified parameter settings layers (L=12), hidden layers (H=128) and attention heads (A=12) totally around 110 M parameters or weights.Using annotated data, classification model is fine tuned with the weights of pre-trained mBERT model. Classification model is fine-tuned with BERT model with following parameters (i.e) batch size of 32, fine tuned for 3 epochs with learning rate as 2e-5 using sentence classification dataset.

IndicBERTmodel[y]is utilized from IndicNLPsuite with specified parameter settings (i.e) max_sequence_length is 128, learning rate is 2e-5,batch size is 32 and fine-tuned for 3 epochs. Compared to mBERT model the utilization of Indian language dataset is high.

Results

- ***Distractor Generation*:** Distractor generation is evaluated with test data which is shown in Table 2 using different Generation methods.
- ***Expert based Evaluation:*** In terms of reliability and plausibility the baseline system and our two step pipeline process are evaluated. We follow evaluation method which is very similar to Chinese fill-in-the-blank distractor method.

Table 2: Validation accuracy of different models

Sentence Validation	Accuracy
mBERT	75.5%
IndicBERT	77.3%

For experiments,55 Tamil fill-in-the-blank questions are considered as test dataset from the annotated datasets (Tamil textbooks). For these 55 Tamil fill-in-the-blank questions, the choices are generated by word2vec model and plausible distractors are refined through context aware models. Human experts proficient in Tamil grammar were used in evaluation process. Experts assessed plausibility of distractor generation on a three point scale, "plausible"(3), "somewhat plausible"(2), or "obviously wrong"(1). Experts assessed the plausibility of a distractor which is generated by baseline and the proposed method. In Table 3 plausibility scores are calculated within the three point scale.

Table 3. Plausibility of the various distractor generation methods

Method	Plausibility	
	Three Point Scale	Avg score
Candidate Collection Method	Somewhat Plausible	1.26
mBERT Model	Strongly Plausible	**2.80**
IndicBERT Model	Strongly Plausible	2.73

5. Conclusion

In this work, plausible affix-based distractors are generated through sentence classification models.The pretrained context-aware language model parameters are fine-tuned for sentence classification task. The contextual information provides more than one morphological form of the keyword within the cloze style questions. Evaluations showed that IndicBERT model outperforms the mBERT model in sentence validation tasks due to huge amount dataset used in IndicBERT compared to mBERT.

Notes:

[x]. https://github.com/AI4Bharat/indic-bert

[y]. https://github.com/google-research/bert/blob/master/multilingual.md

References

Aldabe, I., De Lacalle, M. L., Maritxalar, M., Martinez, E., &Uria, L. (2006, June). Arikiturri: an automatic question generator based on corpora and nlp techniques. In *International Conference on Intelligent Tutoring Systems* (pp. 584-594). Springer, Berlin, Heidelberg.

Chao-Lin Liu, Chun-Hung Wang, Zhao-Ming Gao, and Shang-Ming Huang. 2005. Applications of Lexical Information for Algorithmically Composing Multiple-Choice Cloze Items. In Proc. 2nd Workshop on Building Educational Applications Using NLP, pages 1–8

Chia-Yin Chen, Hsien-Chin Liou, and Jason S. Chang. 2006. FAST: An Automatic Generation System for Grammar Tests. In Proc. COLING/ACL Interactive Presentation Sessions

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ellis, D. G. (1999). *From language to communication*. Routledge.

Juan Pino, M. Heilman, and Maxine Eskenazi. 2008. A Selection Strategy to Improve Cloze Question Quality. In Proc. Workshop on Intelligent Tutoring Systems for Ill-Defined Domains, 9th International Conference on Intelligent Tutoring Systems.

Kakwani, D., Kunchukuttan, A., Golla, S., Gokul, N. C., Bhattacharyya, A., Khapra, M. M., & Kumar, P. (2020, November). inlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings* (pp. 4948-4961).

Keisuke Sakaguchi, Yuki Arase, and Mamoru Komachi. 2013. Discriminative Approach to Fill-inthe-Blank Quiz Generation for Language Learners. In Proc. ACL.

Meisel, J. M. (2011). *First and second language acquisition: Parallels and differences*. Cambridge University Press.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Padamala R. (2013). Issues in syntactic parsing for modern Tamil. In https://shodhganga.inflibnet.ac.in/handle/10603/193795.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., &Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Pino, J., &Eskenazi, M. (2009). Semi-automatic generation of cloze question distractors effect of students' l1. In *International Workshop on Speech and Language Technology in Education*.

Pires, T., Schlinger, E., &Garrette, D. (2019). How multilingual is Multilingual BERT?.*arXiv preprint arXiv:1906.01502*.

Sankaravelayuthan Rajendran. (2015). A Comprehensive Study of Word Fomation in Tamil. 10.13140/RG.2.1.2363.0805.

Shu Jiang and John Lee. 2017. Distractor Generation for Chinese Fill-in-the-blank Items. In Proc. 12th Workshop on Innovative Use of NLP for Building Educational Applications, page 143–148.

Simon Smith, P. V. S. Avinesh, and Adam Kilgarriff. 2010. Gap-fill Tests for Language Learners: Corpus-Driven Item Generation. In Proc. 8th International Conference on Natural Language Processing (ICON).

Soricut, R., &Och, F. J. (2015). Unsupervised morphology induction using word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 1627-1637).

Susanti, Y., Tokunaga, T., Nishikawa, H., &Obari, H. (2018). Automatic distractor generation for multiple-choice English vocabulary questions. *Research and Practice in Technology Enhanced Learning*, *13*(1), 15.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... &Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

Yeung, C. Y., Lee, J. S., & Tsou, B. K. (2019). Difficulty-aware Distractor Generation for Gap-Fill Items. In *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association* (pp. 159-164).

Noun Phrase Coreference Using Tree CRFs

Vijay Sundar Ram R. and Sobha Lalitha Devi

AU-KBC Research Centre,

MIT Campus of Anna University,

Chennai, India

October 1, 2019

Abstract

Noun Phrase Coreference is a part of Coreference resolution. This is the task of identifying a noun phrase referring to another noun phrase. A noun phrase can be referred by an acronym, alias or a shorten noun phrases or by the relation words. This paper presents an analysis on the types of corefering noun phrase (NP) pairs and tried to analyse the features suitable for identifying various corefering NP pairs using Tree CRFs. NP Corefereing is used in many NLP applications such as machine translation, information extraction, etc. We have tested the algorithm with English and Tamil data. The results obtained in both the languages are encouraging.

Introduction

Coreference resolution is the task of identifying which noun phrase or mentions refer to the same real-world entity in a text or dialogue. This resolution of entities is required in all major NLP applications such as Question Answering system, information extraction, information retrieval and summarization etc. Coreference resolution mainly has the following subtasks; pronominal resolution, noun phrase coreference resolution and clustering to form the chains for each entity. Here we are going to look deep in noun phrase coreference resolution.

Noun phrase coreference resolution is the task of identifying noun phrase to its antecedent noun phrase. A noun phrase can be referred by an acronym, alias or a shorten noun phrases or by the relation words. Noun phrase resolution and the coreference resolution task got geared up in the last decade. Most of the works in the last decade are machine learning based approaches. In this work, we have used tree Conditional Random Fields (CRFs) for the noun phrase coreference resolution task and heuristic rules are used to cluster and form entity chains.

Literature Survey

The research in the task of referential entities has started from late 70's with Hobb's non- naïve approach for anaphora resolution, using semantic information (Hobb, 1978). The initial researches were

mostly on pronominal resolution. There were two kinds of approach namely knowledge rich and knowledge poor. Initial machine learning approaches, were performed by Dagan and Itai (1990) where they did an unsupervised learning approach for anaphora resolution. Aone and Bennt (1995) and McCarthy et al (1995) used decision tree, a supervised machine learning algorithm for anaphora resolution.

Researchers have focused on improving the coreference chains by improving the noun phrase coreference resolution. Soon et al (2001) has presented a work on noun phrase coreference resolution using decision tree approach, in which he had used 12 features that were learned from the corpus. Cardia and Wagstaff (1999) considered noun phrase coreference resolution as a clustering task. They came up with an unsupervised algorithm, flexible for co-ordinating the application of context-independent and context dependent constraints and preferences for accurately partitioning the noun phrase into coreference clusters. Vincent Ng and Cardia (2002) has investigated on the anaphoricity of noun phrase, and approaches to find the anaphoric and non-anaphoric noun phrases. Yannick Versley has used maximum entropy model to find weights for the hard and soft constraints in finding noun phrase coreference resolution in German newspapers. Stoyanov et al (2010) has presented the various levels of challenges in noun phrase coreference. He has discussed issues from named entity task to coreference revolver. Vincent Ng (2010) has presented a survey report on noun phrase coreference resolution. He has broadly classified the works as mention-pair model, entity-mention model and ranking model. He has presented the merit and disadvantages in finer level. He has also discussed about the knowledge sources used and the evolution metrics used.

There are number of works using supervised machine learning approach for coreference resolution, where both pronominal and noun phrase coreference is addressed. The various works are presented in the table 1.

Machine Learning Approach	Coreference System
Decision Tree	Soon et al (2001), Strube
TiMBL and Ripper	Daelmas and Van den Bosch, Hoste, Hendrix (Dutch) (2008), Ng and Cardie (2002), Martha Recesans (Spanish) (2009)
Conditional Random Fields	McCallum et al (2006), Li et al (2008), Lalitha Devi et al (2011a), Vijay Sundar Ram (2012)

TABLE 1: List of Coreference system and machine learning techniques used

Noun Phrase Coreference Resolution

Noun phrase resolution or Non-pronominal resolution is the task of identifying all the NPs/mentions that refers to the same entity in a text or dialogue. This includes named entities, definite descriptions that refer each other. In the present work we discuss on improving noun phrase coreference resolution. Here we have used a machine learning approach; tree Conditional Random Fields, for identifying the noun phrase coreference pairs.

Description on the Coreference Tagged Data

For English, We have used the conference tagged corpus distributed in the CoNLL Shared task 2011 (Pradhan, 2011). In this work, we have considered Broadcast News (BN) and News Wire (NW) genres. And for Tamil we have used an inhouse developed data. The data was developed using 1000 News genre webpages. The following table 2 presents the percentage of Anaphoric Noun Phrases (NPs) in both the genres in training data.

S.No	Language	Genre	Percentage of AnaphoricNPs (%)
1	English	NW	25.92
2	English	BN	31.69
3	Tamil	News	29.45

TABLE 2: Percentage of Anaphoric Noun Phrases

In the prior works, it has been shown that the NP coreference resolution is mostly dependent on the string match feature, acronyms (Soon et al. 2001).

To analyse the complexity in the NP coreference resolution, we tried to analyse the NP pairs used in building the coreference chains. On analyzing the pairs, we were able to come up with six

different types of NP pairs, namely, complete string match, partial string match, definite NPs, Person NPs, Acronyms and relation words.

The noun phrase reappears exactly in complete string match; the noun phrase reappears as a part of the actual NP in the partial string match. Definite NPs and Person NPs are similar to partial string match. In definite NPs, the NPs are coreffered with a definite description. Example: Aeroflot Airlines -> the airlines. Acronyms are of the type, where the NP is coreffered by its acronym. The last type of NP pairs is the type, where both the NPs will have no string match. These NPs will have relation relations, whole and part relation, relations obtained from definite description, and other relation relations. Example: Mumbai -> the city

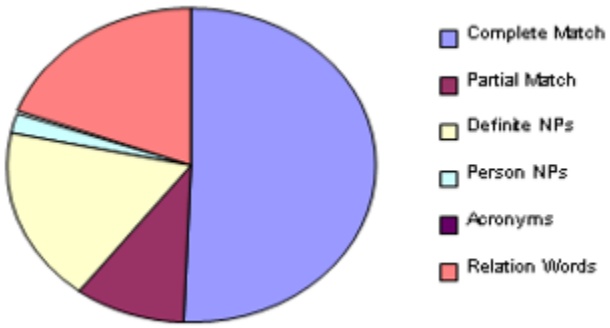


FIGURE 1: NW Genre, English: Classification of NP pairs based on the types of String match

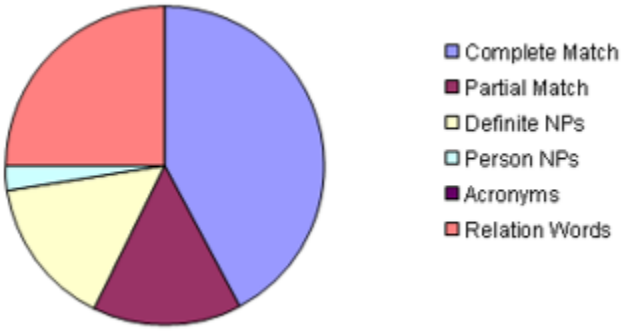


FIGURE 2: BN Genre: Classification of NP pairs based on the types of String match

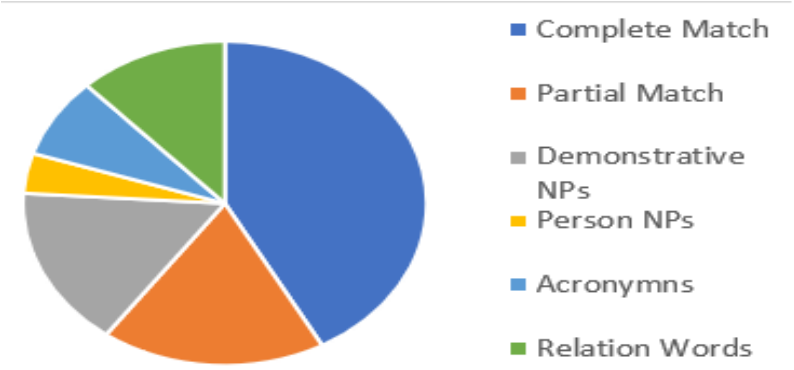


FIGURE 3: News Genre, Tamil: Classification of NP pairs based on the types of String match

In Tamil, we do not have definite NPs as we have in English. But we have usage of demonstratives such as 'itha', 'antha' followed by Noun phrases. We have considered these as demonstrative NPs.

The Fig1, Fig2 and Fig3 shows the distribution of types of NP pairs in building the NP coreference data. Fig 1 and Fig 2 shows the distribution in two different genres in English and Fig3 shows distribution of NPs in Tamil. As focused by the earlier works, the string match plays a vital role as almost 50% NP pairs in NW genre and little less than 50% NP pairs in BN genre and in Tamil text has complete string match. The rest 50% of the NP pairs makes this a challenging task. In the next section, we have described how treeCRFs is used for this task.

Tree CRFs

Tree CRFs are suitable for applications where multiple output labels are needed to be tagged. Sutton et al presented a multiple labelling task of tagging the POS tags and Chunk task using Tree CRFs, where he has explained about the joint prediction of the two labels (Sutton, 2006). Following this work, Tree CRFs was used for Semantic Role Labelling task by (Shilpa Arora (2008), Cohn Philip, Moreau (2009)). Sobha et. al. (2011b) have used Tree CRFs for pronominal resolution task.

A CRFs is called a general CRFs or a tree CRFs, when a general graph is used instead of the linear chain. Here joint prediction is done over multiple output labels. The parameter learning is done using

tree based reparameterization algorithm (TRP), which includes Belief Propagation (BP). This performs exact computation over spanning trees and it updates the suffix to transmit information globally throughout the graph. This helps to have better convergence properties than pure local BP updates (Wainwright 2001).

Tree CRFs forms a clique between the two nodes and tries to predict the labels for these two noun phrases, based on the features used for learning (Bradley 2010). Here we try to make a joint prediction over the corefering NPs.

When a cliques C= {y_c, x_c} are given, CRFs define the conditional probability of a state assignment

$$p(y|x) = \frac{1}{Z(x)} \prod_{e \in C} \phi(x_e, y_e)$$
$$\phi(x_e, y_e) = exp \sum_k \lambda_k f_k(x_e, y_e)$$

given the observation set.

Here Φ (x_c, y_c) is a potential function. And Z(x) is the partition function. We have used GRMM: A Graphical Models Toolkit (Sutton 2006).

Features Used in Tree CRFs

It is well known that the feature selection is the most important task in machine learning technique. As described in section 2.1, we need to come up with features to learn three types of corefering NP pairs. Here we have come up with general features and specific features.

The various features used in Tree CRFs training are presented in the following table 3.

1	**General Features**	
1.1	*Positional Feature*	
1.1.1	sentence initial position	1 if the NP_i occurs in the starting of the sentence else 0, 1 if the NPj occurs in the starting of the sentence else 0
1.1.2	After PP	1 if the NP_i occurs after a preposition else 0, 1 if the NPj occurs after a preposition else 0
1.1.3	End of the sentence	1 if the NP_i occurs in the ending of the sentence else 0, 1 if the NPj occurs in the ending of the sentence else 0
1.2	*NP Type*	
1.2.1	definite NP	1 if the NP_i starts with 'the' else 0, 1 if the NPj starts with 'the' else 0

1.2.2	demonstrative NP	1 if the NP_i starts with demonstratives such as 'that', 'this', 'these' else 0, 1 if the NPj starts with demonstratives 'the' else 0
1.2.3	Proper Name	1 if NP_i is a proper noun, else 0, 1 if NP_j is a proper noun, else 0
2	**Specific Features**	
2.1	*Complete Match*	
2.1.1	Full String Match	1 if NPi and NPj has full string match, else 0
2.1.2	Alias	1 if NPi is an Alias of NPj or vice-versa, else 0
2.1.3	Appositive	1 if NPi , NPj has appositive relation, else 0
2.2	*Partial Match*	
2.2.1	Percentage of Match	percentage of string match between NPi and NPj
2.2.2	Distance	number of lines between NPi and NPj
2.2.3	Capital Letter	1 if NP_i is in capital letter, else 0, 1 if NP_j is in capital letter, else 0
2.3	*Relation Words*	
2.3.1	head Noun	head Noun in NPi and NPj
2.3.2	distance	number of lines between NPi and NPj

TABLE 3: Feature Set used in Tree CRFs learning

For Tamil data, we used the same set of feature except feature number 2.2.3, as there is capital letter in Tamil. For preprocessing Tamil text, we have used a morphological analyser built using rule based and paradigm approach (Sobha et al. 2013). PoS tagger was built using a hybrid approach where the output from Conditional Random Fields technique was smoothened with rules. (Sobha et al. 2016b). Clause boundary identifier was built using Conditional Random Fields technique with grammatical rules as features (Ram et al. 2012). Named Entity built using CRFs with postprocessing rules is used (Malarkodi and Sobha, 2012). Table 4 show the precision and recall of these processing modules.

S.No	Preprocessing Modules	Precision (%)	Recall (%)
1	Morphological Analyser		95.61
2	Part of Speech tagger	94.92	94.92
3	Chunker	91.89	91.89
4	Named Entity Recogniser	83.86	75.38
5	Clause Boundary Identifier	79.89	86.34

TABLE 4: Shallow Parser modules and their performance scores.

Experiment, Results and Evaluation

We have evaluated NP coreference resolution engine for English and Tamil text. For English, we used the development data provided in NW and BN Genre of CoNLL shared task 2011. BN Genre data had 782 corefering NP pairs and NW Genre data had 1898 corefering pairs. For Tamil we used the 1000, online Tamil News articles. The overall performance of corefering NP pair identification is presented in the table 5.

S.No	Language	Genre	Precision %	Recall %
1	English	BN	79.03	69.43
2	English	NW	68.88	73.71
3	Tamil	News	81.14	66.67

TABLE 5: Performance of corefering NP pair identification

To analyse the performance of the identification, we started to explore the recall in each type of corefering NP pairs. The detailed recall in each type of corefering NP pairs is given in table 6 and table 7 for BN and NW genre data in English respectively and table 8 for News genre text in Tamil.

Type of Corefering NP pair	Number of pairs present	Number of pairs identified	Recall %
Complete Match	298	298	100
Partial Match	104	102	98
Definite NPs	114	89	76.72
Person NPs	9	9	100
Acronym	2	2	100
Relation Words	253	43	16.5

TABLE 6: Detailed analysis of corefering NP in BN Genre, English

Type of Corefering NP pair	Number of pairs present	Number of pairs identified	Recall %
Complete Match	908	908	100
Partial Match	168	149	88.69
Definite NPs	269	249	92.56
Person NPs	20	20	100
Acronym	8	7	87.5
Relation Words	525	66	12.5

TABLE 7: Detailed analysis of corefering NP in NW Genre, English

Type of Corefering NP pair	Number of pairs present	Number of pairs identified	Recall %
Complete Match	876	876	100

Partial Match	172	153	88.95
Demonstrative NPs	279	251	89.96
Person NPs	18	18	100
Acronym	11	9	81.82
Relation Words	223	79	35.42

TABLE 8: Detailed analysis of corefering NP in News Genre, Tamil

On analyzing the table 6, 7 and 8, it is very evident that the machine learning system fails to capture the relation between two NPs, where there is no string match. The Complete Match NPs are identified properly. On analyzing the precision of the system, the system has identified the Complete Match NP pairs with high accuracy, where false positive is high in Partial Match and Definite NPs. Though two NPs, which have partial match between them, can possibly be a corefering noun phrase, it is not true in all case. There are a large number of non corefering NP, which have partial match between them. And partial matching NPs may refer two different entities, example “Municipal committee” and ‘Advisory Committee”. These bring more number of false positives. Noun-Noun anaphora resolution engine fails to handle NPs as given in example 1, as in Tamil we do not have definiteness marker, these NPs occur as common noun. Consider the following discourse.

Ex.1.a

maaNavarkaL pooRattam katarkaraiyil
Student(N)+Pl demonstration(N) beach(N)+Loc
nataththinar.
do(V)+past+3pc
(The students did demonstartions in the beach.)
Ex.1 .b
kavalarkaL maaNavarkaLai kalainthu_cella
Police(N)+Pl students(N) disperse(V)+INF
ceythanar.
do(V)+past+3pc
(The police made the students to disperse.)

Consider the discourse Ex.1. Here in both the sentences ‘maaNavarkaL’ (students) has occurred referring to the same entity. But these plural NPs occur as a common noun and the definiteness is not

signalled with any markers. So we have not handled these kinds of definite NPs which occur as common nouns.

Popular names and nicknames pose a challenge in noun phrase coreference resolution. Consider the following examples; ‘Gandhi’ was popularly called as ‘Mahatma’, ‘Baapuji’ etc. Similarly, ‘Subhas Chandra bose’ was popularly called as ‘Netaji’, ‘Vallabhbhai Patel’ was known as ‘Iron man of India’. These types of popular names and nick names occur in the text without any prior mention. These popular names, nick names can be inferred by world knowledge or deeper analysis of the context of the current and preceding sentence. Similarly shortening of names such as place names namely ‘thanjaavur’ (Thanajavur) is called as ‘thanjai’ (Tanjai), ‘nagarkovil’ (Nagarkovil) is called as ‘nellai’ (Nellai), ‘thamil naadu’ (Tamil Nadu) is called as ‘Thamilagam’ (Tamilagam) etc introduce challenge in noun phrase coreference identification. These shortened names are introduced in the text without prior mention. The other challenge is usage of anglicized words without prior mention in the text. Few examples for anglicized words are as follows, ‘thiruccirappalli’ (Thirucharapalli) is

anglicized as ‘Tirchy’, ‘thiruvananthapuram’ (Thiuvananthapuram) is anglicized as ‘trivandrum’, ‘uthakamandalam’ is anglicized as ‘ooty’. Spell variation is one of the challenges in noun-noun anaphora resolution. In News articles, the spell variations are very high, even within the same article. Person name such as ‘raaja’ (Raja) is also written as ‘raaca’. Similarly, the place name ‘caththiram’ (lodge) is also written as ‘cathram’. In written Tamil, there is a practice of writing words without using letters with Sanskrit phonemes. This creates a major reason for bigger number of spell variation in Tamil. Consider the words such as ‘jagan’ (Jagan), ‘shanmugam’ (Shanmugam), and ‘krishna’ (Krishna), these words will also be written as ‘cagan’, ‘canmugam’ and ‘kiruccanan’. These spell variations need to be normalised with spell normalisation module before pre-processing the text.

Even without utilizing any knowledge resources, we are able to fairly identify the corefering NP pairs. By using resources such as WordNet, thesaurus, we can handle the NP pairs, which does not have

string match. A dictionary of popular names, related words and spell verient words will be very useful. More syntactic features should be used in identifying correct Partial match pairs and disambiguation the non-corefering pairs.

Conclusion

The paper describes the identification of corefering NP pairs using tree CRFs, which is part of Coreference resolution task. Here we have analysed the possible types of corefering NP pairs and their distribution in the corpus. We have used the coreference tagged CoNLL share task 2011 data for train and testing for English and inhouse developed News genre Tamil data. On the whole the machine learning algorithm performs well. The machine learning algorithm works badly for NP pairs, where there is no string match. These can be improved by adding features from other knowledge resources. The false positive that occurs with partial match NPs requires more features to disambiguate the corefering and non-corefering pairs.

References

Aone, C. and Bennett, S.(1995). Evaluating automated and manual acquisition of anaphora resolution strategies. In: 33rd Annual Meeting of the Association for Computational Linguistics, pp. 122--129.

Cardie, C. and K. Wagstaff. (1999). Noun phrase coreference as clustering. In 1999SIGDAT Conference on Empirical Methods in Natural Language Processing and Very LargeCorpora, College Park, Md. pages 82-89.

Dagan, I. and Itai, A. (1990) Automatic processing of large corpora for the resolution of anaphora references. In 13th Conference on Computational linguistics, Vol. 3, Helsinki, Finland, pp. 330--332.

Fei, Li., Shuicai, Shi., Yuzhong, Chen., and Xueqiang, Lv. (2008). Chinese Pronominal Anaphora Resolution Based on Conditional Random Fields. In International Conference on Computer Science and Software Engineering, Washington, DC, USA. pp. 731--734

Hendrickx, I., Hoste, V. and Daelemans, W. (2008). Semantic and syntactic features for Dutch coreference resolution. In A. Gelbukh (Ed.), CICLing-2008, Vol. 4919 LNCS, Berlin, pp. 351--361. Springer Verlag.

Hobbs, J. (1978). Resolving Pronoun References. Lingua 44, 339--352.

Joseph K. Bradley. and Carlos Guestrin. (2010). Learning Tree Conditional Random Fields. In: 27th International Conference on Machine Learning.

Malarkodi, CS. Pattabhi RK Rao & Sobha Lalitha Devi (2012). Tamil NER – Coping with Real Time Challenges. In Proceedings of Workshop on Machine Translation and Parsing in Indian Languages, COLING 2012, Mumbai, India.

McCarthy, J. F. and Lehnert, W. G. (1995). Using decision trees for coreference resolution. In C. Mellish (Ed.), Fourteenth International Conference on Artificial Intelligence, pp. 1050--1055

Moreau, E. and Tellier I. (2009). The crotal SRL system: a generic tool based on tree-structured CRF. In: Thirteenth Conference on Computational Natural Language Learning: Shared Task, pp 91--96

Ng, V. and Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In. 40th Annual Meeting of the Association for Computational Linguistics, pp. 104--111

Ng, V. (2010). Supervised Noun Phrase Coreference Research : The First Fifteen Years. In: ACL 2010. pp. 1396--1411.

Pradhan, S. and Ramshaw, L. and Marcus, M. and Palmer, M. and Weischedel, R. and Xue, N. (2011). CoNLL-2011 Shared Task: Modeling Unrestricted Coreference in OntoNotes. In Fifteenth Conference on Computational Natural Language Learning: Shared Task, pp 1--27, Portland, Oregon, USA

Recasens, M., Hovy, E. (2009) A Deeper Look into Features for Coreference Resolution. Lalitha Devi, S., Branco, A. and Mitkov, R. (eds.), Anaphora Processing and Applications (DAARC 2009), LNAI 5847: 29--42. Springer-Verlag Berlin Heidelberg. (2009)

Shilpa Arora, Frank Lin, Hideki Shima, Mengqiu Wang, Teruko Mitamura, Eric Nyberg. (2008). Tree Conditional Random Fields for Japanese Semantic Role Labeling. Unpublished Manuscript

Sobha Lalitha Devi, Pattabhi RK Rao, Vijay Sundar Ram, Malarkodi, C. and Alikadeswari, A. (2011a): Hybrid Approach for Coreference Resolution. In: Fifteenth Conference on Computational Natural Language Learning: Shared Task, pp 93--96, Portland, Oregon, USA

Sobha Lalitha Devi, Vijay Sundar Ram and Pattabhi RK Rao. (2011b). Resolution of Pronominal Anaphors using Linear and Tree CRFs. In. 8th DAARC, Faro, Portugal

Sobha Lalitha Devi, Marimuthu, K. Vijay Sundar Ram, R. Bakiyavathi, T. and Amudha, K (2013). Morpheme Extraction in Tamil using Finite State Machines. In Proceedings of Morpheme Extraction Task at FIRE.

Sobha Lalitha Devi, Pattabhi RK Rao and Vijay Sundar Ram, R (2016b). AUKBC Tamil Part-of-Speech Tagger (AUKBC-TamilPoSTagger 2016v1). Web Download. Computational Linguistics Research Group, AU-KBC Research Centre, Chennai, India

Soon, W., H. Ng, and D. Lim. (2001). A machine learning approach to coreference resolution of noun phrases. Computational Linguistics 27 (4), 521--544.

Stoyanov, V., Cardie, C., Gilbert, N., Riloff, E., Buttler, D., Hysom, D. (2010). Coreference Resolution with Reconcile. In ACL pp 156--161

Sutton, C.: GRMM: A Graphical Models Toolkit. http://mallet.cs.umass.edu.2006.

Sutton, C. and McCallum, A. (2006). An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar (eds), Introduction to Statistical Relational Learning. MIT Press.

Vijay Sundar Ram and Lalitha Devi. S. (2012), "Coreference Resolution using Tree-CRF", In A. Gelbukh (ed), Computational Linguistics and Intelligent Text Processing, Springer LNCS Vol. 7181/2012 pp 285 – 296

Vijay Sundar Ram, Bakiyavathi, T. Sindhujagopalan, Amudha, K. and Sobha, L. (2012). Tamil Clause Boundary Identification: Annotation and Evaluation. In the Proceedings of 1st Workshop on Indian Language Data: Resources and Evaluation, Istanbul.

Wainwright, M. J., Jaakkola, T. and Willsky A.S. (2001): Tree-based reparameterization for approximate inference on loopy graphs. In: NIPS, pp. 1001--1008

என் பேரு தமிழு (en̲ pēru tamiḻu): A Speech Recognition Robot for Tamil

Vasu Renganathan
University of Pennsylvania
vasur@sas.upenn.edu

0. Introduction

Speech recognition devices have been very popular in the recent years and they have reduced much of human's mechanical routines with simple and confined speech expressions. Amazon's Alexa, Google's 'Hi Google', Apple's Siri etc., are some of the popularly used tools that have been in the market for quite some time. Many of these devices recognize English speech considerably well within a limited structure with the basic idea that speech can be accounted for when it is within a pre-defined and a well-controlled construction. "Alexa! Set the alarm at 4:30PM", "Hi Google! What's the time now?", "Alexa! What is the weather now?", "Alexa! What is the status of the Flight AI177" etc., are some of the expressions which these gadgets can respond to in a very live manner. These devices when connected to online resources such as weather stations, atomic clocks, airline schedules and so on, they become very convenient in accessing information conveniently. When posed with some random expressions like, "Alexa! Why am I having a headache?", "Hi Google! Do I have COVID?", "Alexa! Why do I feel cold?" etc., they attempt to parse some of the key words such as "headache", "COVID", "COLD" respectively and come up with related answers to these key words as they can access from the web resources. These represent attempts to process any manageable structures from a mechanical perspective, without any human intelligence or rigorous Natural Language processing tasks involved. Thus, it can be assumed that leveraging the natural language utterances to the extent possible is one of the goals of this type of research and building such devices may not require much of AI techniques at all. Despite such simplicity in technology, this type of devices is not yet available for any Indian languages, including Tamil, although it is quite possible to build one by employing these technologies.

Along these lines of research, an attempt is made in this work to build a mobile Robot called தமிழு (tamiḻu), which can recognize a set of pre-recorded speech in Tamil and respond accordingly as coded in the algorithm written in C++ and Python. This is not precisely like any speech recognition devices as discussed above, which can parse words and syntactic structures of a language to certain extent. But, it is a pattern recognition tool which can match pre-recorded speech patterns to that of a similar one as uttered by the users. It is speaker centric in the sense that only the voice of those who recorded the speech can be recognized and responded, but not any random voice of such patterns. Its scope is very limited and mostly narrowed down to some of the scopes within a pedagogical context in that it can be used to train language learners to proceed through their carefully pre-recorded voice to casual spoken speech. However, a routine is provided in the system to replace the existing recordings with the voice of other users, so anyone can attempt to use this device as needed. Unlike the devices such as Alexa, Google, Sri etc., this is a mobile Robot which can be given command to move around with the four wheels and recognize objects on the way so it can divert its moving direction. This device is built with a speech recognition card and Arduino's technology. (Cf. https://www.arduino.cc).

என் பேரு தமிழு (en̲ pēru tamiḻu) – A Tamil Robot.

The Robot that is discussed here consists within it a C++ code interacting with the pre-recorded Tamil sentences, as stored in the speech recognition board, and the Arduino's resources to operate the motors as attached in the device; pinging any website to play audio as desired; responding to custom-tailored speech and so on. Although the potentials of this device can be expanded infinitely with other possibilities, particularly using the built-in Unix operating system with Wifi along with some of the potentials of the versatile software such as Python, PHP and the databases like MySQL, only a few built-in functionalities are discussed here. As this research is still in progress more features are being added upon on an ongoing basis to make it more robust for any imaginable practical applications. A set of template videos, mostly developed toward language learning purposes, is demonstrated at http://robot.tamilnlp.com and subsequently described in detail in this article.

0.1. Moving around with L298N Motor Drive Controller and Tamil Commands:

The L298 motor drive controller as attached to Arduino boards can be used to make mobile robots and this technology is made use of in the Tamil Robot with a driving factor being a set of Tamil commands interacted through the speech recognition module. The commands such as "**முன்னால வாங்க**" (mun̲n̲āla vāṅka)[1], "**பின்னால போங்க**" (pin̲n̲āla pōṅka), "**சுத்துங்க**" (cuttuṅka), "**எடது பக்கமா சுத்துங்க**" (eṭatu pakkamā cuttuṅka), "**வலது பக்கமா சுத்துங்க**" (valatu pakkamaa cuttuṅka) etc., are some of such pre-recorded speech patterns in the recognition board and they can be used to operate the Robot to move in different directions such as forward, backward, left, and right. When the board's speech recognition microphone coupled with the C++ code's loop structure, a sequence of commands can be given to the Robot to move around accordingly. This feature can be used conveniently by the learners of Tamil to get used to directions in Tamil and subsequently be able to operate the Robot to move around with their own voice. As mentioned earlier, this Robot is built in such a way that anyone can replace the existing commands with their own voice and subsequently make the Robot follow their own directions. A module to give directions and getting used to corresponding direction related expressions in Tamil is demonstrated in the video http://robot.tamilnlp.com/directions.mp4.

0.2. Repeat until you learn: Learning Tamil literature with the Tamil Robot.

With the use of the in-built functions to play sound in Arduino and correspondingly with the use of Python's sound libraries, it becomes possible to play pre-recorded speech efficiently in a number of different ways. 'Repeating the lines of verses until one learns them fluently' is a method that is employed to train students learn Tamil literary verses such as **ஆத்திசூடி** (āṭṭicūṭi) and **புறநானூறு** (puṟanāṉūṟu) to certain extent. Some of these poems are recorded with each line of the verses as separate recognition unit, and subsequently the students are capable of repeating

[1] Polite expressions are used to address this Robot for two reasons. One, for using a gender-neutral expression and for the other to make unambiguous utterances to make the recognition easier for the Robot. In fact, nothing prevents the user from replacing them with their own expressions of choice, using the corresponding routine but by taking a careful measure to avoid any ambiguity for the pattern matching process by the Robot.

them by one line at a time, as played by the Robot. When the student utters a line that is recognizable by the Robot, it plays the subsequent line for the student to repeat, so the student can repeat the same line, or the line next to it. Thus, when the student utters the following line, instead of repeating the same line, the Robot continues to utter subsequent lines successively. Thus, the Robot and the student get to recite the entire poem with each of them saying a line at a time, a stage which can be conveniently called a "fully memorized stage" for students. This method of "Repeat until you learn" is demonstrated in the videos http://robot.tamilnlp.com/aattucudi.mp4 and http://robot.tamilnlp.com/sangam.mp4, with lines from Atticudi and Purananuru respectively. This method can be made use of to memorize poems from any Tamil literary piece, provided the lines of the poems are recorded in the speech recognition board ahead of time and made available for the Robot to recognize. This method can also be used to train students learn and memorize words such as Tamil months, days, Tamil years and related others, which require a constant practice. For instance, the video http://robot.tamilnlp.com/tamil_months.mp4 demonstrates as to how Tamil months are practiced with a student either repeating after what is said by the Robot or alternate with Robot to utter them all.

0.3. Live connection with online database servers.

Not many Arduno's boards come with built-in Wifi and operating systems, except for the board Yun (https://www.arduino.cc/en/Guide/ArduinoYun). What is unique about Yun is that it has a built-in full-blown Linux operating system along with all the Wifi capabilities. With Linux operating system, it is virtually possible to make use of all the functionalities of any mini-computer including to run software applications such as Python, PHP, Shell Script and so on; and interact with databases as needed. Raspberry PI also has similar capabilities, and it is quite possible to use the speech recognition card with Raspberry PI, but it hasn't been attempted yet in this work. Not to mention the fact that a wide variety of audio and video libraries can also be made use of with the Arduino and Speech recognition boards provided they are linked with the Yun board. Exploiting these functionalities of Yun, this Robot interacts with an online database dynamically and play audio files from English-Tamil pedagogical dictionary and Thirukkural verses stored in audio format. With the commands such as, "தமிழு திருக்குறள் சொல்லுங்க" (tamiḻu tirukkuṟaḷ colluṅka), "அகராதிச் சொல் சொல்லுங்க" (akarātic col colluṅka), the Robot picks verses from Thirukkural and words from dictionary randomly and plays it for the user. Although this system is not furnished with many other internet based features such as querying weather, flight schedules etc., as Alexa and other devices do, it is quite possible to enable such features in this system by using the corresponding APIs. However, with an ongoing update of the online databases for Tamil, it is quite possible to make the Robot serve new content dynamically without having to make any change in the Robot itself. What it entails is that Robot on the one hand, voice commands and interaction with the database servers on the other hand make a live interaction with the knowledge content as stored in the databases.

0.4. Interactive conversational partner.

Conversations between Robot and the user can be made possible provided a set of dialogues are written ahead of time and subsequently one part of the dialogue is pre-recorded in the recognition card and the other part is communicated by the user. Every dialogue consists of human's turn versus Robot's turn and the human's turn should be recognizable by the Robot as

pre-recorded in the recognition board. Hence, all the human's turns when recognized, the Robot plays the corresponding response in a real-time mode. Such conversations need not necessarily be performed in a sequence from beginning to end and the Robot can be made to respond to any part of the conversation at any given time. A sample interactive conversation dialogue is made use of with the Tamil Robot and demonstrated in the video: http://robot.tamilnlp.com/beginners_tamil.mp4. Given the fact that there are complex expressions in Tamil which are dificult to learn, a separate dialogue is written to particularly practice a number of patterns involving many complex grammatical constructions in Tamil. For example, expressions like பார்த்துக்கிட்டிருக்கிறதற்காகவாவது (pārttukkiṭṭirukkiṟataṟkākavāvatu), வறேண்ணுட்டாங்களா? (vaṟēṇṇuṭṭāṅkaḷā?), போயிடமுடியாதுங்றதுக்காகவா (pōyiṭamuṭiyātuṅṟatukkākavā) are some of the commonly used complex expressions by native speakers of Tamil in daily routines but are very hard to learn and utter by any second language learner. When such expressions are recorded and kept in the recognition board, the Robot may be made to recognize such expressions from the student and respond accordingly, in the same technique of 'repeat until you learn'. Robot repeats the same expression until it is uttered correctly by the student and does not proceed to subsequent expression unless it is pronounced correct. A sample video along this line of method is demonstrated in http://robot.tamilnlp.com/learn_to_speak.mp4.

0.5. Talking Robots

With a sequence of commands and responses being the driving factor in constructing these Robots, it is surely possible to let the Robots interact each other provided the question/answer sessions are planned carefully with appropriate time intervals. An attempt is made to make two Tamil Robots interact each other with simple conversation and it is demonstrated in the video http://robot.tamilnlp.com/tamil_sangam.mp4. Although it is quite possible to make a long conversation between Robots along these lines, such effort has not been attempted yet. The following four key steps would help understanding the process involved in the performance of this Robot.

1) Robot Listening Mode → Audio Signal ← Users Issuing Commands

2) Pattern Matching with Pre-Recorded Dynamically Replaceable Tamil Commands
3) Perform an action with a match
(or)
4) Go to the listening mode with a signal for failure.

Step 3 can be one of the actions as outlined in sections 1.1 through 1.5, and the listening mode and issuing commands as in step 1 should be concurrent and happen simultaneously. It is to be noted that while performing the action as in step 3, the listening mode and subsequently issuing commands would be stopped, unless it is done in a loop so issuing commands can be done while performing an action by the Robot. Thus, the part of issuing commands can be performed by a human or by another Robot, provided the commands are issued carefully synched with the listening mode.

0.5.1. Commands and Responses:

The recognition board consists of twelve units and each unit can hold within it thirty unique commands, with a total of three hundred and sixty commands. The commands as stored in each unit should be unique in nature and should not be ambiguous in any manner for a good result. Special care needs to be taken to make the commands without any ambiguities involving identical words and expression. Each time a recognition is made within a single unit with what is uttered corresponding action is performed by the robot. Separate command needs to be included in each unit to switch to other units to perform actions within them. For example, the first unit has the trigger word **வணக்கம்** 'vaṇakkam'. Upon recognizing this command, the other commands as stored within the first unit can be accessed. Subsequently to switch between other units, appropriate command needs to be included in each unit. To cite an example, when the command **ஆத்திசூடி சொல்லுங்க** 'ātticūṭi colluṅka', is recognized, the unit with the commands for **ஆத்திசூடி** 'ātticūṭi' will be called upon and subsequently all the actions related to it can be performed within it. Similarly, when the command **புறநானூறு சொல்லுங்க** 'puṟanāṉūṟu colluṅka' is recognized, it will be switched to the unit where **புறநானூறு** 'puṟanāṉūṟu' verses are stored and correspondingly all responses related to it can be engaged in. When commands like **பாடுங்க** 'pāṭuṅka', **எம்ஜியார் பாட்டு பாடுங்க** 'emjiyār pāṭṭu pāṭuṅka' etc., are recognized, corresponding audio files are played using Python's audio library. Thus, recognition and corresponding actions are compartmentalized within the commands as stored within twelve units and interactions between human and robot within each of these units should be well planned ahead of time as to how they are accessed. In this sense, the Robot is made to act upon within a set of twelve domains of speech situations, and switching between these domains need to be determined ahead of time. Perhaps, it might be possible to write a machine learning algorithm to switch between units by the Robot itself based on the users multiple responses, but that sort of attempts have not been made yet in this system.

0.6. Conclusion

It is quite possible to let a robot understand and process commands of any complex type but doing so would require a very sophisticated and complex natural language processing technique built as part of the code. Particularly, such attempts should involve within it many NLP processors such as text to speech, speech to text, morphological and syntactic analyzer and so on, but such attempt is yet to be made in this work. However, the intricacies and processes involved in building such NLP methods have been studied and demonstrated already on various other contexts including text to speech, morphological and syntactic analyzer etc., as can be seen in the works of Renganathan (2016, 2014, 2001). What is demonstrated in this work is a unique pattern matching method constructed within a set of Tamil speech patterns. All possible scopes of this work are demonstrated with a set of sample videos as filmed in a live mode with the Tamil Robot called **தமிழு** (tamiḻu).

1. References[2]

Similar technological systems consulted:
Amazon's Alexa - https://en.wikipedia.org/wiki/Amazon_Alexa.
Google Assistant - https://assistant.google.com.
Apple's Siri - https://www.apple.com/siri/

Renganathan, Vasu (2016). Computational Approaches to Tamil Linguistics, Cre-A. Chennai, India.
Renganathan, Vasu (2016). Computational Approaches to Tamil Linguistics: Scopes and Prospects. In the proceedings of the 15th Tamil Internet Conference, Gandhigram Rural University, Dindigul, Tamil Nadu. (http://www.uttamam.org/papers/16_02.pdf).
Renganathan, Vasu (2014). Computational Phonology and the Development of Text-to-Speech Application for Tamil. In the Proceedings of the International conference on Tamil Internet, 2014, Pondicherry, India. (http://www.uttamam.org/papers/14_35.pdf).
Renganathan, Vasu (2001). Development of Morphological Tagger for Tamil, In the Proceedings of the International Conference on Tamil Internet 2001, Kuala Lumpur, Malaysia. (http://www.uttamam.org/papers/01_34.pdf).

[2] The online resources as referred to in this article were accessible on 11/24/2021.

மாநாட்டு நிகழ்வின் இணை ஒருங்கிணைப்பாளர்கள்

Central Institute of Indian Languages, Mysuru

Tamil Virtual Academy

University of Hyderabad

Chennai Institute of Technology

Bharathiyar University, Coimbatore

இணையம் வளர்க்கும் தமிழ்

www.tamilinternetconference.org

இணையத்தில் தமிழ் பரப்புவோம் - மக்கள்
இதயத்தில் தமிழ் நிரப்புவோம்.

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