# Focused Crawler Based on Reinforcement Learning and Decaying Epsilon-Greedy Exploration Policy

Parisa Begum Kaleel
Department of Applied Mathematics and Computational Sciences, PSG College of Technology, India
kpb.amcs@psgtech.ac.in

Shina Sheen
Department of Applied Mathematics and Computational Sciences, PSG College of Technology, India
ssh.amcs@psgtech.ac.in

**Abstract:** *In order to serve a diversified user base with a range of purposes, general search engines offer search results for a wide variety of topics and material categories on the Internet. While Focused Crawlers (FC) deliver more specialized and targeted results inside particular domains or verticals, general search engines give a wider coverage of the web. For a vertical search engine, the performance of a focused crawler is extremely important, and several ways of improvement are applied. We propose an intelligent, focused crawler which uses Reinforcement Learning (RL) to prioritize the hyperlinks for long-term profit. Our implementation differs from other RL based works by encouraging learning at an early stage using a decaying $\epsilon$-greedy policy to select the next link and hence enables the crawler to use the experience gained to improve its performance with more relevant pages. With an increase in the infertility rate all over the world, searching for information regarding the issues and details about artificial reproduction treatments available is in need by many people. Hence, we have considered infertility domain as a case study and collected web pages from scratch. We compare the performance of crawling tasks following $\epsilon$-greedy and decaying $\epsilon$-greedy policies. Experimental results show that crawlers following a decaying $\epsilon$-greedy policy demonstrate better performance.*

**Keywords:** *Focused web crawling, infertility, information retrieval, reinforcement learning.*

*Received January 30, 2023; accepted June 7, 2023*
*https://doi.org/10.34028/iajit/20/5/14*

## 1. Introduction

A search engine's main objective is to facilitate users' rapid and effective information discovery. A search engine's index of web pages, documents, photographs, videos, and other online content is searched by users who enter specific words, phrases, or queries. The search engine then returns results that are pertinent to the user's query.

Search engine technology has scaled up dramatically to keep up with the expanding amount of information on the Internet. Google and other search engines have the structure and algorithms to handle billions of pages [1], provide huge volume of results which is difficult to choose from. Vertical Search Engines (VSE) [7] are popular, have a structured index and precise than Generalized Search Engine. VSE provides more search capabilities and information extraction. The intelligence behind the success of the VSE is the focused web crawlers or topical crawlers.

Focused Crawlers (FC) or Topical Crawlers are web crawlers that start with a set of seed pages and fetch as many relevant pages as possible while avoiding irrelevant pages. Before fetching the document, these crawlers should be able to forecast that the next Uniform Resource Locator (URL) refers a page relevant to the topic based on:

1. Text similarity between the subject and the anchor text of a page link Classic Focused Crawlers (CFC).
2. Semantic similarity criterion for computing page-to-topic relevance: a page and a topic are relevant if they have the same concept, related phrases, and keywords (Semantic Crawlers).
3. Using a training method for assigning visit priority (Learning Crawlers).

FC have been designed which use machine learning for relevance calculation. Some FC are adopted to extract from the web archives.

The growing use of Internet technology (such as computers and mobile phones) has made health information and support more accessible. 80% of Internet users search for health-related topics online [15]. Web-based resources provide a great advantage over print material and personal discussions. Web resources provide the advantages of availability, instant access to the latest information, and peer group discussions about similar difficulties. The Internet overcomes geographic and socioeconomic obstacles, allowing underprivileged groups, such as infertile couples and linguistic minorities, access health information and support. As per World Health Organization (WHO- ICMART Glossary), infertility is ‘a disease of the reproductive system defined by the

failure to achieve a clinical pregnancy after 12 months or more of regular unprotected sexual intercourse' [3], [24]. According to the National Health Portal of India (NHP), infertility affects up to 15% of reproductive-aged couples globally. According to the WHO, India's overall prevalence of primary infertility ranges from 3.9 to 16.8%. Infertility is a chronic illness that causes people to experience feelings of shame, lack of control, and many emotions [5]. Since it provides a high level of privacy in highly stigmatized situations such as infertility and conception-related issues, the Internet is utilized to learn the medical jargon used in infertility diagnosis and therapy.

Infertile couples, particularly women, search online resources to get infertility issues and other health information [3, 15]. In general, web crawlers like Google are used as the primary source of information by couples before seeking professional help. The number of indexed pages in Google is 5.76 billion. Out of these pages, only a fraction of them will be related to the fertility domain. Distressed patients may be actively searching for information related to their healthcare needs, but their specific needs might not always be met through conventional search engines. Search of health information in Internet, how to search and their success is discussed in [11]. Hence, we need a focused crawler for infertility domain that could provide information from healthcare-related websites, support groups, forums, and other resources that cater to distressed patients' needs. It could provide latest information on treatments, mental health resources etc.

The main contributions of this work are:

- Development of a focused crawler for the healthcare (infertility) domain.
- To propose an improvement in the URL Selection (URL-SL) policy.
- To compare the performance of the different FC, based on various scoring mechanism.

This focused crawler is the first for the infertility domain, satisfying the user needs. Moreover, it fetches highly relevant web pages in a short time.

## 2. Literature Review

The different works in FC are ML based [4], Adopted Focused Crawlers (AFC) to extract from Web archives [8], and Tunnelling based [2, 10]. The first focused or topical crawler is implemented with a classifier and a distiller [4]. Focused crawler with semantic similarity for relevance calculation is discussed in [6]. Crawlers which learn to assign priority value is discussed in [12].

A link which looks less relevant may lead to relevant pages in nearby steps called Tunnelling [2]. Avoiding irrelevant pages is a major concern. RL provides a solution to this issue. Reinforcement Learning (RL) is a machine learning technique that uses a reward function to learn optimal decision-making via rewards or penalties [23].

RL based FC are discussed in [9, 10, 22]. RL is modelled as Markov Decision Process (MDP) with action values stored in tabular form because of smaller search space in [21]. RL with Gradient Descent (GD) and Function Approximation (FA) with deriving only the parent score is discussed in [9]. A topic specific crawler built on web page classification and Link Priority Evaluation (LPE) based on Content Block Partition (CBP) is discussed in [14]. A heuristic based strategy, selective use of link context is applied to improve topical crawling [13]. Focused crawling using RL is applied for Classifier selection among Neural Network (NN), Support Vector Machines (SVM), and Naïve Bayes (NB) and Decision Trees (DT) [19]. RL is applied to focused crawling along with linear FA and GD in [10], compared the synchronous and asynchronous methods of updating the action values.

Incremental online learning is utilized to improve the performance of the framework proposed in [22] which reduces data bias in the test data. A bandit-based selection method is used to choose the highest scoring host and an online classifier picked a connection from the host in [16]. Bandit-based selection leads to an over valuing of short-term behaviour and a larger impact of variance leading to false conclusions.

A hybrid Stochastic Gradient Descent-Particle Swarm Optimisation (hybrid GPSO) technique is used in conjunction with a metaheuristic algorithm to optimise web crawling and choose more relevant web sites for the crawler to get [20].

The agent in the RL environment must strike a balance between greedily exploiting what has been learned thus far to select the links that give bigger rewards in the short term, and continuously investigating (exploring) the environment that is the dynamic web to gain additional information and perhaps attain long-term gains. Extensive research has been done to determine the optimum ways to balance exploitation and exploration [16, 17, 18]. The $\epsilon$-greedy technique for the exploration/exploitation technique is utilized in [10]. The agent normally selects a random URL $\epsilon$ times and selects an URL with high value 1- $\epsilon$ times. The $\epsilon$-greedy algorithm discovers the best action early, but it keeps looking. It tries to explore even after crawling so many pages and gaining experience, leading to sub-optimal pages, which is a disadvantage.

Limitation of $\epsilon$-greedy technique for the exploration/exploitation technique motivated us to improve the performance of the focused crawler. The learning agent is encouraged to explore more during the initial stages of crawling and acquire URLs with relevant content. Exploring new URLs at the starting periods and gradually moving to exploiting the maximum priority value link is being done through the decaying $\epsilon$-greedy to improve the performance. Here $\epsilon$

is decayed and hence at the later stages where ϵ is less, high priority link will be selected.

## 3. Background

Basic crawlers recursively followed the hyperlinks between web documents to collect the most web pages possible. Only the linkage structure of the web is considered, not the page content. Hence a need for focused crawlers arose to limit the concentration to the topic of interest. The functionality of the basic web crawler, focused web crawler and learning crawlers (RL-based crawler) are discussed.

Basic Web Crawler: The starting URLs will be loaded in the URL queue (Frontier). It takes one URL at a time from the frontier. Next, the URL's webpage is downloaded. All links on the page are extracted (additional website and page addresses) and added to a queue to be downloaded later. This process will be continued until a particular number of pages are retrieved. The working flow of the crawler is depicted in Figure1. The limitation of this type of crawler is that it retrieves all the pages it encounters, which is costly in resource constrained environments.

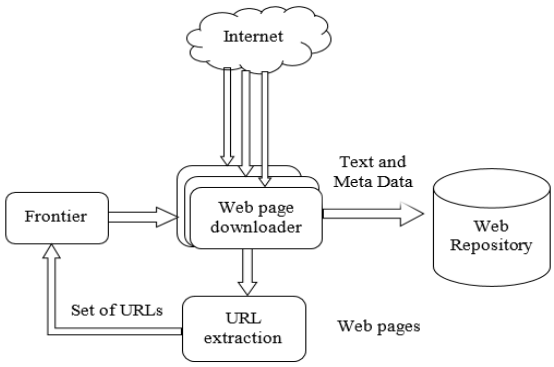



Figure 1. Basic web crawler model.

FC initially will have a certain amount of seed URLs. Each URL in the queue is fetched, the web page downloaded and passed to a Parser and Extractor component which extracts all the links from the web page. Next, the relevance calculator component calculates the relevancy of the web page. The web page is saved in the Relevant Pages database if relevant, and the extracted URLs are added into the frontier (priority queue) with this relevance value as the priority. The process continues until the maximum page limit is reached.

### 3.1. Focused Crawlers

A Focused crawler chooses from the frontier (URL queue), URLs that are more likely to be relevant to a particular topic. This crawler tries to obtain as many relevant sites as possible while avoiding irrelevant pages to conserve network and Central Processing Unit (CPU) resources. Figure 2 depicts the architecture of a basic focused crawler.

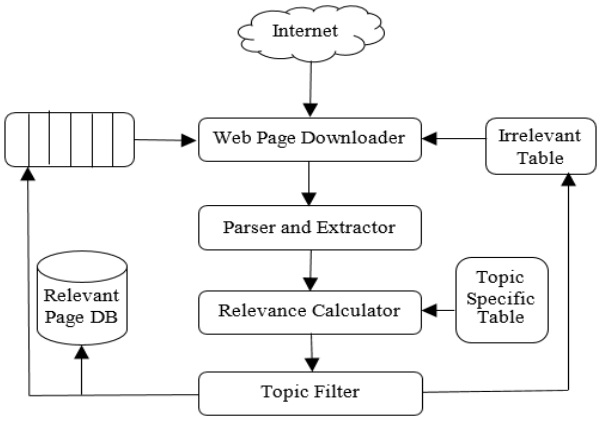



Figure 2. Basic focused crawler architecture.

In basic focused crawling, only pages that are related to a certain topic are evaluated, leaving away irrelevant pages. In the crawling process, pages related to the topic may not be directly connected. An irrelevant web page may link to relevant web pages indirectly after some steps. If a crawler skips a web page considering it irrelevant, the relevant pages that could be obtained through this link are not obtained. Less relevant pages need to be followed to get highly relevant pages called as tunnelling [2]. There should be a limitation on how far this search should continue.

### 3.2. Reinforcement Learning

RL is the process of learning to attain a goal through interacting with the environment. Making a series of decisions is a solid strategy. The agent learns to achieve a goal in an unpredictable and highly complicated environment. The agent receives rewards or punishments for the action it is executing. The agent learns how good it is to take certain action in a given state over time through experiencing actions measured by the value of an action. In the long run, RL strives to maximize the entire payoff. As a result, it is perfect for focused web crawling.

A large portion of the research on RL is based on the idea of the MDP. As stated in [22], the RL problem can be treated as an MDP. MDP is specified as a 4-tuple <S, A, R, T> where S is a set of states, A is a set of actions, R: S x A← R is a reward function and T: S x A x S ← [0, 1] is a transition function. For an action chosen in a specific state, the reward function outputs a single number, the reward. The likelihood of changing from state's' to state s′ when an action is taken is specified by the transition function π for a policy: S→A links states to behaviours that increase the overall reward over time. The agent's objective is to identify an ideal policy π*.

A MDP is a Markov reward process with decisions. It is a tuple <S, A, P, R, γ>

- S is a finite set of states
- A is a finite set of actions
- P is a state transition probability matrix, $P^{a}_{ss'}$= P[$S_{t+1}$ = s′ | $S_t$ = s, $A_t$ = a]

- R is a reward function, $R_s^a = E[R_{t+1} | S_t = s, A_t = a]$
- $\gamma$ is a discount factor $\gamma \epsilon [0, 1]$

State-Action-Reward-State-Action (SARSA) an algorithm for learning a Markov decision process policy is used in the RL area of ML. SARSA is a FA and Temporal Difference (TD) learning on-policy method. Han *et al*. [10] employed RL for the crawling procedure. Web pages are considered as a set of states denoted by the letter 'S', while direct hyperlinks on a web page are treated as an action set represented by the letter 'A'. When the crawling agent selects and follows a hyperlink, it moves from the current web page to the next connected page. The relevancy of the linked page to the target topic is assessed. The relevance value of the linked page to the specified topic is 'r' in R. From the newly visited page, the agent chooses a hyperlink with a higher estimated value. State-action space is reduced by using a generalization function and updated the value functions using linear FA to tackle the scalability issue.

## 4. Proposed Architecture

An overview of the proposed focused crawler is depicted in Figure 3. The following are the main components of the architecture: Frontier for the storage of Seed URLs, URL-SL, Crawler, Parser, Page Relevance (PR) Calculator, URL Relevance (UR) Calculator, and RL-based Learning Component (RL-LC) for reward calculation and action value updation. Each component is discussed in detail.

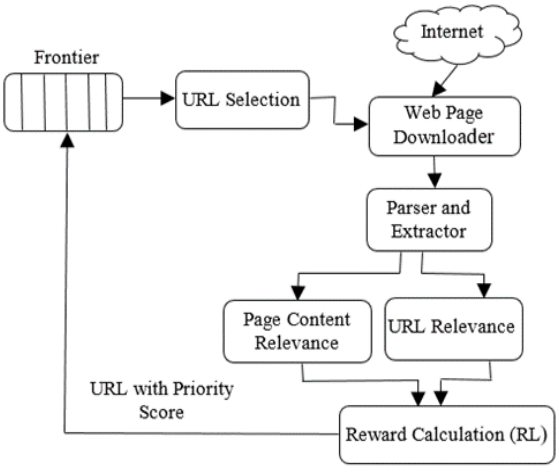



Figure 3. Proposed RL-based focused crawler architecture.

### 4.1. Frontier

The fertility domain seed pages are added to the frontier list. At each iteration, a single link is taken from the frontier and sent to the crawler. The web page is fetched and processed to extract hyperlinks. The priority value for each link is calculated and added to the frontier. The frontier is usually a max-heap priority queue that always returns an item with the highest priority value. As per the algorithm, a random pick of the URL or max priority URL needs to be fetched. Hence it is implemented as a sorted list which enables to make the selection easier. The priority value for the seed URLs are calculated as follows: The web page for the URL is downloaded, and the web page features and action features extracted are used to calculate the cosine similarity value with the feature vector of the topic domain. The initial Q-value is calculated as the inner product of the weight vector 'wt' and the state-action feature vector 'x' as given by Equation (1). This Q-value is used as the priority value for the link and added to the frontier.

$$\hat{q}(st, act, wt) = wt^T x(st, act) = \sum wt_i x_i(st, act) \quad (1)$$

Where $x_i$ *(st, act)* is a function $x_i$: $S \times A \rightarrow R$, its value is a feature of *st* and act.

### 4.2. URL Selection

The most common method of selecting a hyperlink from the frontier, which is employed by many FC, is to choose the link with the highest priority value (Greedy algorithm). The greedy algorithm makes the most of the data by focusing the search on the most promising places in the immediate area. As a result, other good or better links are missed. Always exploiting a dynamic environment like the web may result in a suboptimal approach. Exploration, or picking any connection at random, is also required to find superior links. There should be a balance between exploration and exploitation.

In the Greedy method, only the max priority value links are chosen. Hence, after some crawling, only sub-optimal links are available. In the $\epsilon$-Greedy method, with a random probability $\epsilon$ exploration of links is selected and 1-$\epsilon$ times, exploitation of links is chosen as used by [10]. The issue in this method is that after learning/crawling many pages, exploration is done $\epsilon$ fixed times, under-utilizing the available high-priority links.

The decaying $\epsilon$-greedy method starts with a high value near 1 for the $\epsilon$ parameter. After that, $\epsilon$ diminishes till it fades. This method works as over time, $\epsilon$ gets reduced, and the learning agent becomes more confident of the optimal action and hence could reduce the exploring. The decaying functions could be based on time step, reward-based, etc., In the URL- SL process, during the later stages, exploitation should be encouraged.

### 4.3. Fetcher and Parser

The fetcher component crawls and fetches the input URL's web page, and this web page content is passed to a parser. Parser extracts required data after removing the Hypertext Markup Language (HTML) tags. A web page typically consists of a large number of URLs. The relevant URLs should be placed at the top of the page [1]. A parameter that could be customized is the number of top links to be fetched from a particular page. A small routine does the work of choosing only the top 'n' links.

### 4.4. URL Relevance Score Calculation

An URL is given to a crawler to crawl and obtain the content. If the focused crawler can predict the relevancy of the URL instead of getting the page and determining its relevance, it can avoid exploring so many irrelevant pages. If the URL of a web page fits the standards for Search Engine Optimization (SEO), it will appear at the top of the search engine results. Regarding URL building, Enge *et al*. [7] suggest the following:

- URL should contain a precise word or term related to the page's content.
- Keywords should be included in URLs.
- To use hyphens to separate words. (On parsing the URL, the words could be extracted for the information retrieval process).

An URL has only a limited amount of text, and hence web page authors aim to include as many relevant words as possible in the URL link text. The URL will be tokenized and after the stop words are removed, a large number of domain related words will be available.

The relevance score of an URL is calculated using Baye's theorem. Baye's theorem provides a way to calculate the probability of a piece of data whether it belongs to a particular domain, using the prior knowledge. A set of URLs which are relevant to the particular domain is used as reference. The relevance score of an URL is calculated using Equation (2).

$$p(Relevance\,|URL) = \frac{p(URL|Relevance) * p(Relevance)}{p(URL)} \quad (2)$$

Where *p(Relevance|URL)*, represents the posterior probability that calculates the probability of relevance of an *URL*, given the *URL*.

- p(URL|Relevance), is the likelihood of the current URL belonging to the Relevant Set.
- p(Relevance) is the prior probability of retrieving a relevant URL.
- p(URL) represents how often we see this URL in the Corpus.

### 4.5. Model of the Proposed Architecture

- **States:** Each hyperlink 'act' represents an action, and each web page is considered a state 'st'. As a result, a huge state-action pair will be available in the online context. State-action pairs are represented in MDP as a table. Policy learning is complex with this representation. As a result, the state-action pairings are generalized.

Web pages and the next hyperlink could be characterized by some features. Hence pages with the same feature values are treated the same, and the assumptions apply to the hyperlinks. Relevance to the target topic is considered a feature of a web page. Relevance is computed as the cosine similarity between the web page vector and the domain subject vector.

- **Actions**: A web page's hyperlinks are also abstracted with some attributes that specify actions. The anchor text's relevance to the target topic and the URL text's relevance to the target topic are both considered features for the actions.

In reference to Equation (1), in the focused crawling scenario, the action-value function is approximated by the linear combination of the feature vector x(st, act) with the weight/parameter vector 'wt'. The feature values $x_i$(st, act) used here are:

- $x_1$(st, act) is a function returning the relevance score of a web page's content.
- $x_2$(st, act) is a function returning the relevance score of the anchor text.
- $x_3$(st, act) is a function returning the relevance score of the URL text.

Algorithm (1) describes RL for the focused crawling algorithm with decaying $\epsilon$-greedy algorithm. Seed URLs are given in order to start the crawling process. All the outlinks from the seed pages are represented by (st, act) pairs on the frontier. The decaying-$\epsilon$ greedy policy is used to extract a link from the frontier. The agent chooses a link greedily 1- $\epsilon$ times and selects a URL at random $\epsilon$ times. The web page is retrieved, and the feature value is set. The new state's reward and feature value are calculated, and the new state's relevance is returned. The weight vector is changed as shown in Equation (3) based on these reward and feature values.

$$wt_{t+1} = wt_t + \alpha[rw_{t+1} + \gamma\hat{q}(st_{t+1}, act_{t+1}, wt_t) - \hat{q}(st_t, act_t, wt_t)]\nabla\,\hat{q}(st_t, act_t, wt_t)] \quad (3)$$

The new weight vector is used to update the new actions obtained from the current state. In such a case, the frontier will have action values calculated at different times, generating an error in the selection. The TD Error, also known as the Bellman error, is the difference between two estimates of two separate time sequences, as shown in Equation (3). The current estimate $\hat{q}(st, act, wt)$ is updated toward the new target reward $r + \hat{q}(st', act', wt)$. The weight 'wt' is modified as specified in Equation (3) to balance the influence of action-values updated at distinct time steps.

*Algorithm 1: RL-based Focused Crawling with Dacaying $\epsilon$-Greedy Policy*

*Data: Seed URLs, Page Limit numPagesLimit, Epsilon Values limit minEpsilonValue, maxEpsilonValue*

```
procedure RLFOCUSEDCRAWL ()
    Set the weights of the value function wt ∈ d
    FRONTIER← Null
    while StartUrls is not empty do
        lnk ← Choose(StartUrls, 1)
        st ←Crawl the page pointed by 'lnk' and parse
        LinkSet ← Collect all the outlinks from 'lnk'
        for all lnk' ∈ LinkSet do
            (lnk', st', act') ←from lnk' find action
```

```
            features of act'
10:         Add (lnk', st', act') to (st', act') pair of
                FRONTIER with initial Q-Value
11:       end for
12:     end while
                        ▷ Decaying ϵ value step function
13:     epsdecrement=(maxEpsilonValue –
            minEpsilonValue) / numPagesLimit
14:     while totalVisitedPages < numPagesLimit  do
15:         if with probability ϵ then
16:             currentItem ← Choose (st, act) from the
                    FRONTIER uniformly at random
17:             select a link (lnk, st, act) from
                        currentItem
18:         else
19:             currentItem←Choose (st, act) having
                    highest  Q-Value from the FRONTIER
20:             select a link (lnk, st, act) from currentItem
21:         end if
22:         if lnk is visited then
23:             continue
24:         else
25:           st' ← Fetch the page and parse (lnk, st, act)
26:             pr ←CosineSimilarityBasedOnTFIDF(lnk)
27:             urlrelscore ← CalculateUrlRelScore(lnk)
28:             rw ←CALCULATEREWARD(pr,
                        urlrelscore)
29:             LinkSet' ← Collect all the outlinks from lnk
30:         end if
31:     end while
32:     for all lnk' Є LinkSet' do
33:         if lnk' is visited then continue
34:         (lnk', st', act') ← Get action features act' of lnk'
35:         end if
36:     end for
37:     if visited page is relevant then
38:             wt_{t+1} = wt_t + α[rw −
                    q̂(st, act, wt)] ∇q̂(st, act, wt)]
39:     else
40:         Choose act' as a function of q^(st', . , wt)
                with ϵ-greedy policy
41:       δ ← rw + γ (st', act', wt) − q̂(st, act, wt)
42:       wt ← wt + α[rw + γ(q̂(st', act', wt) − δ) −
            q̂(st, act, wt)] ∇q̂(st, act, wt)
43:     end if
44:     for all (st, .) pair Є          LinkSet' do
45:         Calculate Q-value of (st', .)
46:         Add (lnk', st', .) to (st',.) pair of FRONTIER with
                Q-value
47:         currentEpsilonValue=currentEpsilonValue -
                epsdecrement
48:         totalVisitedPages = totalVisitedPages + 1
49:     end for
50: end procedure
```

The performances of ϵ-greedy and decaying ϵ-greedy methods of exploration-exploitation are analysed through the different models of focused crawling and scoring mechanisms which are given in Table 1.

Table 1. Focused crawler models and scoring mechanism.

| Model name | Scoring mechanism and URL selection policy |
|---|---|
| **PRGreedy** | PR score only with ϵ-Greedy policy |
| **PRDecay** | PR score only with Decaying ϵ-Greedy policy |
| **URLGreedy** | URL Relevance score only with ϵ-Greedy policy |
| **URLDecay** | URL Relevance score only with Decaying ϵ-Greedy policy |
| **HybridDecay** | Hybrid URL Relevance score and PR score with Decaying ϵ-Greedy policy |

In the PRGreedy model, the reward is calculated using the Cosine Similarity of the page content, and the URLs are chosen using an ϵ -greedy policy.

To calculate the reward, the PRDecay model leverages the Cosine Similarity of the page content, and a decaying ϵ-greedy policy was employed to choose URLs.

The URLGreedy model uses the URL relevance score for the rewarding process. A higher reward is supplied if the URL is having a higher probability; otherwise, a minimal or negative reward is given. The ϵ-greedy policy is used to choose URLs.

Model URLDecay is similar to the URLGreedy model, except that the selection policy is decaying ϵ-greedy.

The hybrid model HybridDecay uses a combination of page content relevance score (cosine similarity based on tf-idf) value and the URL relevance score value. Algorithm (2) explains the reward calculation. The values of the pos_reward1, pos_reward2, and pos_reward3 are 30, 25, 20 respectively. By this method, the learning agent will have different ways of learning.

*Algorithm 2: Reward Calculation*

```
1: procedure CALCULATEREWARD (PR, URLRELSCORE)
2:         if pr > maxthreshold and urlrelscore >= 0.50 then
3:             reward ← pos_reward1
4:         else if pr> maxthreshold and urlrelscore < 0.50then
5:             reward ← pos_reward2
6:         else if pr≤ maxthreshold and urlrelscore ≥0.50 then
7:             reward ← pos_reward3
8:         else if pr≤ maxthreshold and urlrelscore < 0.50 then
9:             reward ← -1
10:        end if
11: end procedure
```

## 5. Data Acquisition and Experimental Set-Up

This section covers the methods of data collection, experiment setup (values of the parameters used), and several evaluation metrics for measuring the effectiveness of the focused crawler for the collection of fertility related web pages.

### 5.1. Data Acquisition

Seed URL Collection: The richness of links in the specific topic being searched determines the performance of a focused crawler, and focused crawling relies on a general web search engine for starting points. Seed selection significantly influences the crawling efficiency. A suitable hub website should be directed by a seed URL (points to many good pages). The URLs can be collected by querying the general search engines like Google or can use the already available major sources to collect URLs like Directory Mozilla Open Directory Project (DMOZ-ODP) which is currently managed by curlie.org, Yahoo! Directory, etc. But getting an equal

proportion of URLs containing fertility hospital information, Artificial Reproduction Techniques (ART), is not feasible from the above said collection.

Fertility hospital websites provide information about the ART, doctors team, the experience of the doctors and the reviews of the benefited patients, etc., Govt-managed fertility society websites (available country-wise) provide detailed information on infertility, and a list of accredited hospitals in the country and state. They also project the In Vitro Fertilization (IVF) success rates; embryos transferred statistics and multiple birth statistics. Blogs on infertility provide information about the issues of the infertile patients, their treatments and their success stories, etc., providing moral support. Hence manual collection of URLs was executed. Some URLs belonging to the fertility domains are given in Tables 2 and 3.

Table 2. Authentic websites for infertility information.

| Name of the organization | Website address |
|---|---|
| Society for Assisted Reproductive Technology (SART) | www.sart.org |
| The American Society for Reproductive Medicine (ASRM) | https://www.asrm.org/ |
| European Society of Human Reproduction and Embryology (ESHRE) | https://www.eshre.eu/ |
| Reproductive Medicine Associates of Connecticut (RMACT) | https://www.rmact.com/ |
| Indian Fertility Society | https://www.indianfertility-society.org/ |

The seed URLs given in Table 2 are taken one by one and tested at random for the availability of fertility related webpages. These seed URLs lead to highly relevant websites mostly. Hence a total of 50 seed URLs from various sources like curlie.org, SART, and ASRM are collected, which are major sources of fertility information.

The discount factor ($\gamma$) ranges between [0,1]. It regulates the relative value of future rewards vs. immediate ones. When the discount factor is low, future rewards are viewed as less important, and the agent is more likely to focus on actions that provide immediate rewards.

The learning rate ($\alpha$) or step size determines how quickly the model adapts to the problem. Setting a value that is too little may result in a lengthy training process, while setting a value that is too large may result in learning a sub-optimal set of weights in a short period, as well as an unstable process.

Table 3. Blogs for infertility information.

| Name of the forums, blogs | Website address |
|---|---|
| Reproductive Medicine Associates of Connecticut (RMACT) | https://www.rmact.com/ |
| EGGSPERIENCE | https://eggsperience.com/ |
| Center of reproductive medicine | https://www.infertilitytexas.com/blog |
| Waiting for Baby Bird | https://waitingforbabybird.com/ |

The different parameters, along with their values used in the $\epsilon$-greedy and decaying $\epsilon$-greedy methods, are given in Table 4.

Providing a high value for $\epsilon$ at the initial stages of crawling will give a great chance to explore. The gradual decrease of $\epsilon$ value reduces the count of exploration, helping in utilizing the experience (URLs with max priority value). The minimum value is set as 0.1, since in later stages, a small part of exploration is involved instead of choosing a sub-optimal action in terms of exploitation.

Table 4. Parameters for the $\epsilon$ -Greedy and Decaying $\epsilon$ -Greedy models.

| Parameter name | $\epsilon$ -Greedy | Decaying $\epsilon$-Greedy |
|---|---|---|
| $\epsilon$ | 0.1 | 0.9 - 0.1 |
| $\gamma$ | 0.9 | 0.9 |
| $\alpha$ | 0.001 | 0.001 |

## 5.2. Experimental Setup

All the crawlers discussed here are written in Python and use the BeautifulSoup and Urllib libraries. The performance of the Breadth-First Crawler (BFC) is used as a benchmark. The Cosine similarity of the web page content is used to calculate the PR score. As RL crawlers, it ignores the feature value of the state and action. All links on the current page are added to the frontier with the associated cosine similarity value as a priority score if a web page is crawled and judged to be relevant to the topic domain. Following that, a link from the queue is obtained for crawling.

Limits on connection timeout and downloading time frames are implemented for performance concerns. The crawling operation is continued for the $\epsilon$-Greedy algorithms until the preset number of pages is obtained, which in this case is 10000. The crawling process is continued in Decaying $\epsilon$- Greedy algorithms until the $\epsilon$ decays to a minimum value.

As previously indicated, crawler performance is determined by the percentage of relevant pages downloaded, which is referred to as "Harvest rate". Cosine similarity based on tf-idf is used to score the web page content in this study, and a URL relevance score is employed to know the URL's relevance. The crawler agent earns the greatest reward of 30 if the crawled page has a tf-idf based relevance greater than 0.10 and the urlrelscore is greater than 0.50. "If the crawled page's tf-idf score is larger than 0.10 and the urlrelscore is less than 0.5", the page is still considered relevant, and the reward is set to 25. If the page score is less than 0.10 and urlrelscore is greater than or equal to 0.50, then the reward for the agent is 20, and otherwise it is set as -1.

## 5.3. Performance Measures

The performance of a focused crawler has a direct impact on the crawling. As a result, it is important to track how quickly the crawler filters out the relevant pages. Precision will be the fraction of relevant pages crawled, and recall will be the fraction of relevant pages crawled. Because the relevant set for every issue is unknown on the Internet and hence difficult to quantify,

real recall is difficult to determine. As a result, the harvest rate measure was utilized to assess the concentrated crawler's performance. The following is how the harvest rate is calculated:

The harvest rate is the percentage of relevant online pages for a specific topic that determines how successfully it rejects irrelevant web pages. Harvest rate is calculated as given in Equation (4).

$$Harvest - rate = \frac{\Sigma i \epsilon V \ r_i}{|V|} \quad (4)$$

where $V$ is the number of web pages crawled by the focused crawler, $r_i$ is the relevance of the $i^{th}$ web page against the given topic that takes the value 0 or 1. If it is relevant $r_i$=1, else $r_i$=0.

## 6. Results and Analysis

The performance comparison of the baseline crawler Breadth-First Search (BFS) against the RL-based crawler PRGreedy has been depicted in Figure 4. The total number of relevant pages is almost twice in RL-based crawler compared to the BFS crawler. It means that RL effectively helps in finding relevant pages as time steps increase.

Figure 5 shows the number of relevant pages obtained at different time steps in the crawling process. In this figure, the number of crawled web pages is plotted against the number of relevant pages when the number of crawled pages is N.

At all-time steps of crawling, the hybrid model (HybridDecay) is collecting more relevant pages compared to the other models. The performance is better than other models for less number of crawled pages and for a higher number of relevant pages.

The performance of a focused crawler is best analysed with the help of average harvest rate. For each model, five trials (tasks) are executed. Each task consists of crawling 10000 pages. Figure 6 shows the comparison of the performance of all the five models based on harvest rate. It can be seen from Figure 6 that as the number of crawled web pages increases, the average harvest rates of the five models are falling.

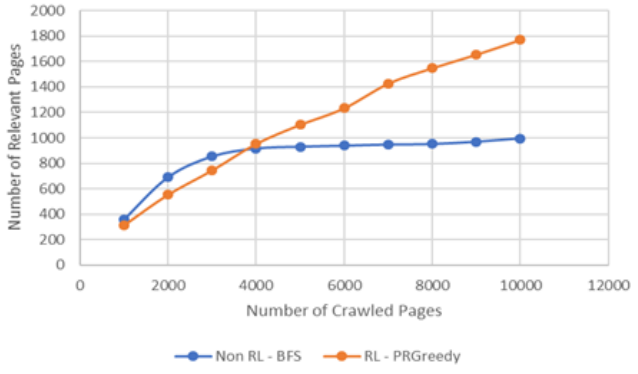



Figure 4. Performance comparison of BFS (Non-RL) crawler against RL Crawler.

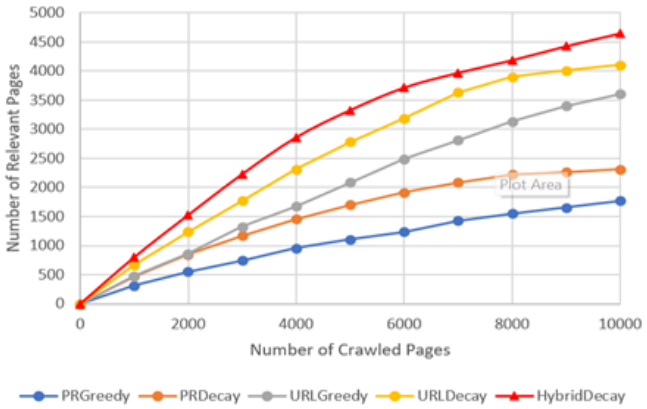



Figure 5. Performance comparison-accumulated number of relevant pages.

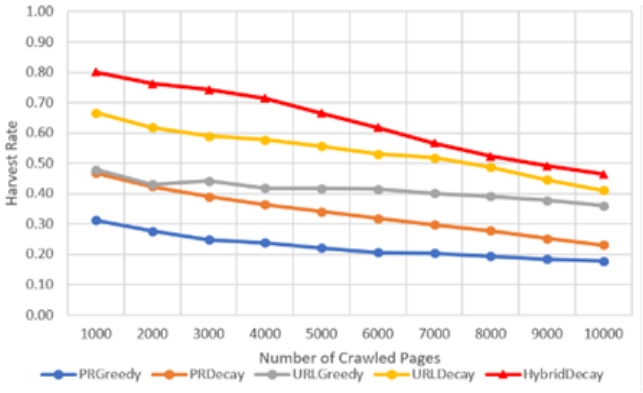
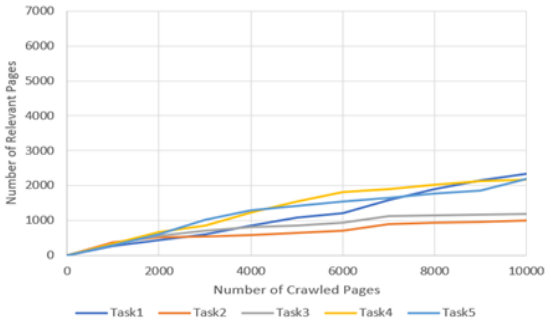


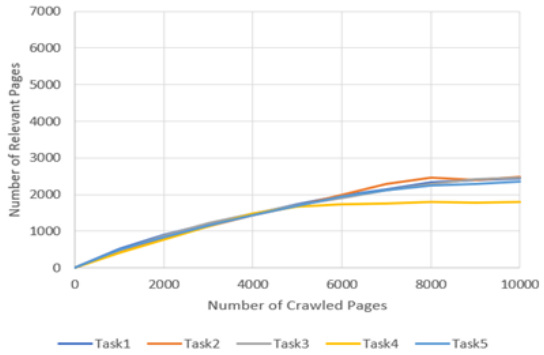

Figure 6. Performance comparison-average harvest rates of all models.

The average harvest rates for the five tasks at the point that corresponds to 10000 crawled web pages are: 0.18, 0.23, 0.36, 0.41, and 0.46. These values indicate that the harvest rate of HybridDecay model is 1.27, 2.00, 2.28, and 2.56 times larger than those of other models. Therefore, the figure indicates that this crawler has the ability to collect more topic-relevant web pages than the other models.

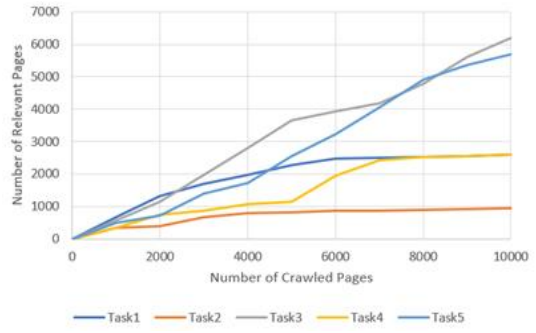
As Figures 5 and 6 indicate, the best average performance is achieved by the hybrid model HybridDecay which used page content score and URL relevance score, following the Decaying $\epsilon$-greedy policy.



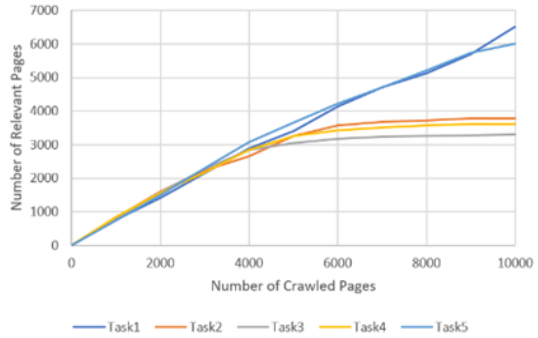

a) PRGreedy.



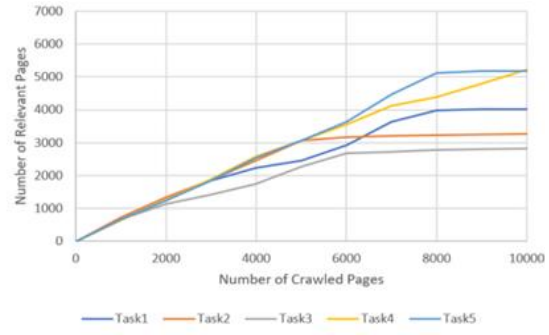

b) PRDecay.




c) URLGreedy.




d) URLDecay.




e) HybridDecay.

Figure 7. Crawling task response of ϵ-greedy and decaying ϵ-greedy policy models.

Figure 7-a)-(e) show the crawling task result for 5 different trials of each model. Figure 7-a) and (c) are the results for the models which follow the ϵ-greedy policy, whereas Figure 7-b), (d), and (e) are the result for the models following the decaying ϵ-greedy policy.

Figure 7-a) for the model using cosine similarity score and ϵ-greedy policy shows the curves for the five different tasks are moving in random. In ϵ-greedy, the ϵ value is fixed, enabling exploration at a constant rate always. Since the ϵ value is fixed, the exploration rate is the same throughout the crawling process. Hence a minimum value like 0.1 is set. So only 10% of the time, the learning agent explores the environment and remaining time it has to be greedy searching the maximum action value function already available leading to sub-optimal actions.

Figure 7-b) shows the crawling tasks with decaying ϵ-greedy policy and cosine similarity score. Since ϵ value starts with a high value, more exploration of the web environment is allowed at the initial crawling stage, and hence a consistent and smooth curve could be seen at the start. Over a period of time, the ϵ value is decayed and exploitation encouraged, leading to random movement of the task curves.

In the same way, Figure 7-c) and (d) show the result of crawling based on the URL relevance score and ϵ-greedy policy against the decaying ϵ-greedy policy. Each task shown as an individual curve has variations from the start of the crawl with ϵ-greedy policy. The number of relevant pages fetched in each crawling task is different, starting from a low of 900 pages and up to 6200 pages. Hence the average number of relevant pages will be low. In the decaying ϵ-greedy along with the URL relevance score, a small variation at the tail end of the crawling process is observed; all the tasks have a minimum of 2800 relevant pages limit. The average number of relevant pages is hence high compared to the ϵ-greedy policy method. Also, in decaying ϵ-greedy policy methods, all trials collect more relevant pages in the starting stage itself.

The crawling result of 5 different tasks of the hybrid model (HybridDecay) is shown in Figure 7-e). In this model, both the PR score and URL relevance score are used to calculate the reward. It follows the decaying ϵ-greedy policy for the exploration and exploitation process. The curves for all the crawling tasks follow the same pattern in the initial stage, and there is not much variation. All the tasks end up in above 3200 relevant pages for a total of 10000 pages crawled. The average number of relevant pages from all the 5 crawling tasks is higher for the HybridDecay model when compared to other models. The web pages which have high page content relevance and the URLs which have more relevance to the target topic are getting a high score which may be selected in near time. This leads to an increase in the number of relevant pages crawled.

- **Comparison with other RL Focused Crawlers**: The proposed RL based focused crawler is compared with other RL based FC, based on the concepts tunnelling, exploration/exploitation, online learning and decaying ϵ-greedy and shown in Table 5.

Table 5. Comparison of RL based FC.

| S.No | Work done by | f1 | f2 | f4 | f3 |
|---|---|---|---|---|---|
| 1 | Rennie and McCallum [21] | - | - | - | - |
| 2 | Pant *et al*. [18] | - | ✓ | - | - |
| 3 | Partalas *et al*. [19] | - | - | ✓ | - |
| 4 | Han *et al*. [10] | ✓ | ✓ | ✓ | - |
| 5 | Proposed Method | ✓ | ✓ | ✓ | ✓ |

- f1-Tunnelling. • f2-Exploration/exploitation.
- f3-Online learning. • f4-Decaying ϵ-Greedy.

Since FC will be part of VSE tunnelling technique will help collect more relevant pages. By selectively following exploration or exploitation the crawler is able to choose from the higher priority value URLs or explore new URLs in the web. As per decaying ϵ-greedy algorithm, the learning agent is encouraged to explore more during the initial stages of crawling and acquire more URLs with relevant content. Exploring new URLs at the starting periods and gradually moving to exploiting the maximum priority value link improves the performance of the crawler. When a crawler is trained offline with a limited number of URLs and their web contents, the crawler may be less efficient when it is implemented for a real web.

## 7. Conclusions and Future Work

This paper focuses on the development of a focused crawler using RL with infertility domain as a case study. An analysis of the performance of the ϵ-greedy and decaying ϵ-greedy policy in the exploration and exploitation process of RL has been done. From the experiments, it is clear that the model utilizing decaying ϵ-greedy policy with a combined relevance of web page content and relevance of URL shows high performance. It collects more relevant pages at the early stage of crawling and is consistent in all trials.

The ϵ-decay strategy followed does not consider the performance of the agent or any feedback from the environment. Instead of assuming that the agent is learning more every episode, we wait for proof of the agent's learning before reducing the ϵ value. This method is called Reward-Based Decay. This decay is not dependent on number of episodes, but on the performance of the agent and hence it is possible for different agents with different learning capabilities to experience a similar degree of exploitation based entirely on their performance.

A prominent challenge in our information age is identifying an URL as relevant to the domain or not. Different methods of URL relevance score calculation could be utilised to improve the learning of the crawler.

## 8. Data Availability

The data that support the findings of this study are available from the corresponding author, upon reasonable request.

## References

[1] Agichtein E., Brill E., and Dumais S., "Improving Web Search Ranking by Incorporating User Behaviour Information," *in Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, pp. 19-26, 2006. https://doi.org/10.1145/1148170.1148177

[2] Bergmark D., Lagoze C., and Sbityakov A., "Focused Crawls, Tunneling, and Digital Libraries," *in Proceedings of the Research and Advanced Technology for Digital Libraries: 6th European Conference*, Rome, pp. 91-106, 2002. https://doi.org/10.1007/3-540-45747-X_7

[3] Brochu F., Robins S., Miner S., Grunberg P., Chan P., Lo K., Holzer H., Mahutte N., Ouhilal S., Tulandi T., and Zelkowitz P., "Searching the Internet for Infertility Information: A Survey of Patient Needs and Preferences," *Journal of Medical Internet Research*, vol. 21, no. 12, 2019. DOI: 10.2196/15132

[4] Chakrabarti S., Van den Berg M., and Dom B., "Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery," *Computer Networks*, vol. 31, no. 11-16, pp. 1623-1640, 1999. https://doi.org/10.1016/S13891286(99)00052-3

[5] Cousineau T. and Domar A., "Psychological Impact of Infertility," *Best Practice and Research Clinical Obstetrics and Gynaecology*, vol. 21, no. 2, pp. 293-308, 2007. DOI: 10.1016/j.bpobgyn.2006.12.003.

[6] Ehrig M. and Maedche A., "Ontology-Focused Crawling of Web Documents," *in Proceedings of the ACM Symposium on Applied Computing*, Melbourne, pp. 1174-1178, 2003. https://doi.org/10.1145/952532.952761

[7] Enge E., Spencer S., and Stricchiola J., *The Art of SEO: Mastering Search Engine Optimization*, O'Reilly Media: Sebastopol, 2015. https://www.oreilly.com/catalog/errata.csp?isbn=9781491948965

[8] Gossen G., Risse T., and Demidova E., "Towards Extracting Event-Centric Collections from Web Archives," *International Journal on Digital Libraries*, vol. 21, no. 1, pp. 31-45, 2020. https://doi.org/10.1007/s00799-018-0258-6

[9] Grigoriadis A. and Paliouras G., "Focused Crawling Using Temporal Difference-Learning," *in Proceedings of the Methods and Applications of Artificial Intelligence: 3rd Hellenic Conference*,

*SETN*, Samos, pp. 142-153, 2004. https://doi.org/10.1007/978-3-540-24674-9_16

[10] Han M., Wuillemin P., and Senellart P., "Focused Crawling through Reinforcement Learning," *in Proceedings of the 18th International Conference on Web Engineering*, Cáceres, pp. 261-278, 2018. https://doi.org/10.1007/978-3-319-91662-0_20

[11] Houston T. and Allison J.., "Users of Internet Health Information: Differences by Health Status," *Journal of Medical Internet Research*, vol. 4, no. 2, pp. 1-10, 2002. DOI: 10.2196/jmir.4.2.e7.

[12] Li J., Furuse K., and Yamaguchi K., "Focused Crawling by Exploiting Anchor Text Using Decision Tree," *in Proceedings of the Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, Chiba, pp. 1190-1191, 2005. https://doi.org/10.1145/1062745.1062933

[13] Liu L., Peng T., and Zuo W., "Topical Web Crawling for Domain-Specific Resource Discovery Enhanced by Selectively Using Link-Context," *The International Arab Journal of Information Technology*, vol. 12, no. 2, pp. 196-204, 2015. https://iajit.org/PDF/vol.12,no.2/6058.pdf

[14] Lu H., Zhan D., Zhou L., and He D., "An Improved Focused Crawler: Using Web Page Classification and Link Priority Evaluation," *Mathematical Problems in Engineering, Hindawi Publishing Corporation*, vol. 2016, pp. 1-10, 2016. https://doi.org/10.1155/2016/6406901

[15] McCarthy D., Scott G., Courtney D., Czerniak A., Aldeen A., Gravenor S., and Dresden S., "What Did You Google? Describing Online Health Information Search Patterns of ED patients and their Relationship with Final Diagnoses," *West Journal of Emergency Medicine*, vol. 18, no. 5, pp. 928-936, 2017. doi: 10.5811/westjem.2017.5.34108

[16] Meusel R., Mika P., and Blanco R., "Focused Crawling for Structured Data," *in Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*, Shanghai, pp. 1039-1048, 2014. https://doi.org/10.1145/2661829.2661902

[17] Pant G. and Menczer F., "Topical Crawling for Business Intelligence," *in Proceedings of the International 7th European Conference on Theory and Practice of Digital Libraries: Research and Advanced Technology for Digital Libraries*, Trondheim, pp. 233-244, 2003. https://doi.org/10.1007/978-3-540-45175-4_22

[18] Pant G., Srinivasan P., and Menczer F., "Exploration Versus Exploitation in Topic Driven Crawlers," *in Proceedings of the 2nd International Workshop on Web Dynamics*, Honululu, pp. 88-97, 2002. http://dblp.uni-trier.de/db/conf/www/webdyn2002.html#PantSM02

[19] Partalas I., Paliouras G., and Vlahavas I., "Reinforcement Learning with Classifier Selection for Focused Crawling," *in Proceedings of the 18th European Conference on Artificial Intelligence*, Patras, pp. 759-760, 2008. DOI:10.3233/978-1-58603-891-5-759

[20] Rajiv S. and Navaneethan C., "Hybrid Gradient Strategies in Event Focused Web Crawling," *ECS Transactions*, vol. 107, no. 1, pp. 1219-1234, 2022. DOI :10.1149/10701.1219ecst

[21] Rennie J. and McCallum A., "Efficient Web Spidering with Reinforcement Learning," *in Proceedings of the 16th International Conference on Machine Learning*, Bled, Slovenia, pp. 335-343, 1999. https://www.researchgate.net/publication/2272137_Efficient_Web_Spidering_with_Reinforcement_Learning

[22] Singh N., Sandhawalia H., Monet N., Poirier H., and Coursimault J., "Large scale URL-based Classification Using Online Incremental Learning," *in Proceedings of the IEEE 11th International Conference on Machine Learning and Applications*, Boca Raton, pp. 402-409, 2012. DOI: 10.1109/ICMLA.2012.199

[23] Sutton R. and Barto A., *Reinforcement Learning: An Introduction*, MIT Press, 2018. https://freecomputerbooks.com/Reinforcement-Learning-An-Introduction.html

[24] Zegers-Hochschild F., Adamson G., De Mouzon J., Ishihara O., Mansour R., Nygren K., Sullivan E., and Van der Poel S., "The International Committee for Monitoring Assisted Reproductive Technology (ICMART) and the World Health Organization (WHO) Revised Glossary on ART Terminology," *Human Reproduction*, vol. 24, no. 11, pp 2683-2687, 2009. https://doi.org/10.1093/humrep/dep343

**Parisa Begum Kaleel** obtained her Master of Computer Applications degree from Anna University in 2007. She has industry experience of 3 years. She is currently working as Assistant Professor in Department of Applied Mathematics and Computational Sciences, PSG College of Technology. She is pursuing Ph.D degree under Anna University, Chennai. Her domain of interest is information retrieval.

**Shina Sheen** Professor and Head in the Department of Applied Mathematics and Computational Sciences of PSG College of Technology, India is currently heading the Cyber Security Research lab and holds the SANS GCIA certification in Intrusion detection. She has two funded projects sanctioned by the Department of Science and Technology and her research is in developing proactive methods for ransomware detection.