ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

*(A High Impact Factor, Monthly, Peer Reviewed Journal)*

*Visit: www.ijirset.com*

**Vol. 7, Special Issue 3, March 2018**

# Examine the Contents of Entire Data Packets Using Content Aware Similarity Search (CASS)

Arati Virachand Boralkar[1], Ashvini Uttam Khatal[2], Neha Raju Vaidya[3]

UG Student, Department of Computer Science and Engineering, Sanmati Engineering College, Washim, Maharashtra, India[1]

UG Student, Department of Computer Science and Engineering, Sanmati Engineering College, Washim, Maharashtra, India[2]

UG Student, Department of Computer Science and Engineering, Sanmati Engineering College, Washim, Maharashtra, India[3]

**ABSTRACT***:* A Content Aware Similarity Search (CASS) is designed to determine the contents of entire data packets as well as header and payload information. The network device comprises of a physical interface for transforming analog network signal into bit streams and vice versa. The bit stream coming from the physical interface is forwarded to a traffic flow scanning processor that may be, but is not essentially, broken up into a header processor and a payload analyse. The header processors determine the header information from each data packet, which is used to find out routing information and session identification. The scanned data packets and the associated conclusions undergoes a quality of service processor which enhances the data packets if necessary and performs traffic management and traffic shaping on the flow of data packets based on contents of the data packets. The payload analyser scans the data packet's payload and pairs the payload against a database of knowledge strings. The payload analyser is able to scan across packet boundaries and to scan for strings of variable and arbitrary length. Once the payload has been scanned, the network device can function on the data packet based on the results of the payload analyser. Experimental results confirmed that users could obtain more suitable and reasonable search results t han with a typical web or map search system.

**KEYWORDS:** Data packets, Analog network signal, Routing information, Session identification, Payload analyzer, Database

## I. INTRODUCTION

During the previous decades, significant progress has been made on extracting features for similarity pursuit and object recognition from functionality-easy information, for example, sound, images, video, and other sensor datasets. Since feature-rich data objects are normally spoken to as high-dimensional feature vectors, similarity search is generally actualized as K-Nearest Neighbor (KNN) or Approximate Nearest Neighbors (ANN) search in high-dimensional feature-vector space. The similarity search has many properties as following: Accurate, Time efficient, Space efficient. In addition, the construction of the index data structure should be quick and it should deal with various sequences of insertions and deletions conveniently. A good search mechanism in an efficient content-based search system for feature-rich data. To start with, it ought to convey list items effectively on large datasets without utilizing much CPU and memory assets. For instance, it ought to have the capacity to migration millions of information articles and data objects in seconds. Second, it should to have the capacity to accomplish high search quality by using advanced element extraction technique. With the fast advance in storage technologies, disks with

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Special Issue 3, March 2018

capacity from hundreds of gigabytes to even several terabytes are now very common in personal computers. As a result, the number of documents stored in the local file system, and functionalities that support effective search for a particular document is of ever growing importance. However, achieving such kind of functionalities turns out to be quite challenging. Unlike in the case of Web search, where powerful ranking schemes such as PageRank [22] and HITS [19] can be employed, there is no computers in similar ways. Therefore, state-of-the-art desktop search engines, such as Google Desktop Search, usually adopt pure text based ranking approaches (e.g., TF-IDF scores) lastly, is the Toolkit for similarity search. This toolkit will be different among the most search toolkits we have right now. The goal is to develop a toolkit that can be used to construct search engines for various data types by plugging in specific data segmentations, feature extractions and distance calculation modules.

## II. METHODS

**1) The Ferret Toolkit:-**

The Ferret toolkit is developed to allow systems builders to construct efficient content- on similarity based search systems for various consignment -rich data types. There are three type of the Ferret Toolkit :

2) Core components - The core similarity search engine, an attribute-based search tool metadata management, and a command-line migration interface are the key elements of the toolkit that are independent on data type.
3) Commercialize components - Data acquisition, and performance evaluation tool provide a set of functions that are commonly needed in a similarity search system, web interface.
4) Users of the toolkit are make able to construct search systems by using toolkit, supplying a small number of data-type specific routines that make up and construct the interface, and make to specification the user interface, performance, and data organization components. These provide systems builders to construct similarity search systems for specific data types.
5) Plug-in components - There are two type of plug-in components: segmentation and feature extraction, and distance functions.

**6) Filtering:-**

The brute-force approach is the most general but inefficient technique. It goes through all data objects and their distance from the query object, and returns the most similar ones. With large data sets using this approach is extremely wasteful and time very strong.

The Ferret process makes the process easier into two steps:

- Filtering step - it fastly filters out “bad” answers and produce a small candidate set
- Similarity ranking step - it uses multi-feature object distance function, computes distance for each candidate object. It also returns n nearest objects
- Criteria in picking candidate objects are following:-
- Has at least one segment that is close enough to one of the top segments of the interrogatory object. By the separating the search process into these two steps, one can use a inclined approximation method in the filtering process to improve search speed and allow the user to apply a sophisticated and may be inefficient ranking step to ensure the search quality. If the candidate set size is small, only a small number of EMD computations are needed in the second step. In addition to speed, the filtering step also provides a natural way to integrate the content-based similarity search engine with an attribute-based search engine.

The second is to produce a small and high-quality candidate set quickly. The goal of filtering is to create a small candidate set quickly that contains most of the similar data objects.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Special Issue 3, March 2018


Fig. (a)Best Networking Tools for Filtering


Fig. (b) Ferret Pro Kit

## III. RELATED WORK

The context-aware system has a general architecture divided into three tiers: context-aware application, context-aware service, and context sensors. It is in efficient interaction method between the intelligence system and the surrounding environment. The context-aware system, as the essential component in Ubiquitous Computing, perceives context changes and adapts accordingly. It provides an important way to improve the system intelligence, and comes up with a accessible .

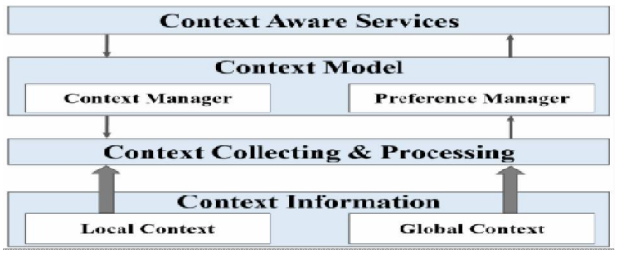


Fig The General Architecture For The Context-Aware System

The context sensors tier refers to generalized sensors containing physical devices or software. Amongst these tiers, the context-aware service tier separates context sensing from application development, providing a universal interface for the higher level to develop context-aware application and support to access diverse sensors in the lower level. The context aware service structure is often subject to a number of factors, such as the user device, whether to support multi-users, or system scalability. Architecture design will vary according to application requirements.

**Context Toolkit architecture:**
Context Toolkit architecture is shown below. It consists of three main components: widget, aggregator and interpreter. Context widgets are charged with retrieving context from sensors. Context aggregators combine many widgets to collect the entire context concerning a given entity. Interpreters provide high-level context to applications based on low-level context from widgets.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Special Issue 3, March 2018

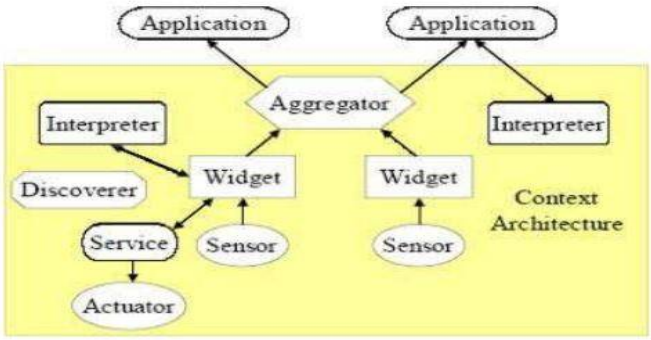


FIG: CONTEXT TOOLKIT ARCHITECTURE

**Advantages:**

i. Provide a sensor package mechanism
ii. Provide a set of APIs, so that data can be accessed through the network context information.
iii. Abstract context data through interpreters.
iv. Provide context of data storage including storage of historical information.
v. Provide basic access control to protect privacy of information

**CASS:-**

The Context-Awareness Sub-Structure middleware (CASS), developed by Trinity College's Patrick, (Fahy and Clarke, 2004), is a server based context-aware middleware. CASS is designed for context-aware application on hand-held computing devices and other small mobile computing devices, supporting low-level sensors and other context-sensitive input devices while at the same time occupying fewer computing resources.

## IV. EXPERIMENTAL RESULTS

- Segmentation - JSEG segmentation tool from UCSB
- Feature extraction - it is 14-d features: 9-d color moments and 5-d bounding box and the segment weight: square root of segment size
- Distance functions - the segment distance: weighted ℓ1distance and the object distance: EMD

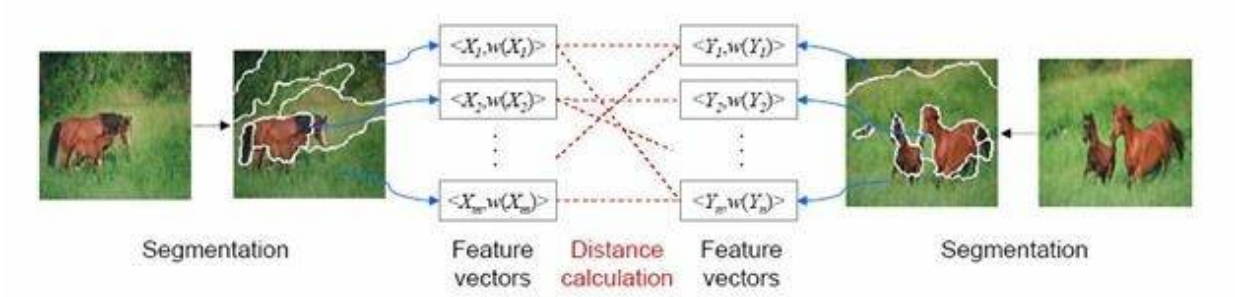


Fig: image similarity search

- Segmentation - Gene expression microarray data: one gene per row
- Feature extraction - Gene expression values

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

*(A High Impact Factor, Monthly, Peer Reviewed Journal)*

*Visit: www.ijirset.com*

**Vol. 7, Special Issue 3, March 2018**

- Distance functions - Pearson correlation, spearman correlation, ℓ1distance

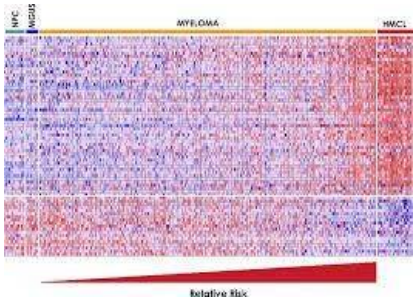


**Fig:** Gene Expression Similarity Search

## VI. CONCLUSION

In the above research paper we can conclude that, as it stands CASS or Content Aware Similarity Search is a promising search algorithm in fact it may even be too ahead of its time due to the fact that some of the searchable items in this algorithm are quite obscure and not applicable to the common person as of this day and age it is not surprising if such an algorithm is adopted by leading search engines in the future.

## REFERENCES

1. Gnome beagle desktop search. http://www.gnome.org/projects/beagle/.
2. H. Cao, H. Hu, D. She0.n, D. Jiang, J. Sun, E. Chen, and Q. Yang. Context-aware query classification. In SIGIR, 2009.
3. H. Cao, D. Jiang, J. Pei, E. Chen, and H. Li. Towards context aware search by learning a very large variable length hidden markov model from search logs. In WWW, 2009.
4. H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click- through and session data. In KDD, 2008.
5. D. Chau, B. Myers, and A. Faulring. What do do when search fails: finding information by association? In CHI, 2008.
6. J. Chen, H. Guo, W. Wu, and W. Wang. imecho: an associative memory based desktop search system. In CIKM, pages 731–740, 2009.
7. J. Chen, H. Guo, W. Wu, and W. Wang. imecho: a context aware desktop search system. In SIGIR, pages 1269– 1270, 2011.
8. J. Chen, H. Guo, W. Wu, and C. Xie. Search your memory! - an associative memory based desktop search system. In ACM SIGMOD, 2009.
9. P. Charta, S. Chita, W. Nejd, and R. Paik. Beagle++: Semantically enhanced searching and ranking on the desktop. In ESWC, 2006.
10. X. Dong and A. Halevy. A platform for personal information management and integration. In CIDR,2005.
11. S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. Robbins. Stuff i' ve seen: a system for personal information retrieval and re-use. In SIGIR, 2003.
12. D. Elsweiler and I. Ruthven. Towards task-based personal information management evaluations. In Proceedings of the international ACM SIGIR conference on research and development in information retrieval, pages 23–30, 2007.
13. J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong. Mylifebits: fulfilling the memex vision. In Proceedings of the ACM Conference on Multimedia, 2002.
14. T. Haveliwala. Topic-sensitive pagerank. In WWW, 2002.
15. T. Joachims. Optimizing search engines using clickthrough data. In ACM SIGKDD, 2002.
16. D. Karger. Haystack: A customizable general-purpose information management tool for end users of semistructured data. In CIDR, 2005.
17. Y. Ke, L. Deng, W. Ng, and D. L. Lee. Web dynamics and their ramifications for the development of web search engines. Comput. Netw. J. (Special Issue on Web Dynamics), 50:1430– 1447, 2005.