ExperTwin: An Alter Ego in Cyberspace for Knowledge Workers

C. Toxtli, C. Flores-Saviaga, M. Maurier, A. Ribot, T. Bankole,
A. Entrekin, M. Cantley, S. Singh, S. Reddy, R. Reddy
Virginia University, Morgantown, WV 26506

***Abstract*—Even though the advent of the Web coupled with powerful search engines has empowered the knowledge workers to quickly find the needed information, it still is a time-consuming operation. Presently there are no readily available tools that can create and maintain an up-to-date personal knowledge base that can be readily consulted when needed. While organizing the entire Web as a semantic network is a long-term goal, creation of a semantic network of personal knowledge sources that are continuously updated by crawlers and other devices is an attainable task. We created an app titled ExperTwin, that collects personally relevant knowledge units (known as JANs) from the Web, Email correspondence, and locally stored files, organize them as a semantic network that can be easily queried and visualized in many formats – just in time – when performing a knowledge-based task. The architecture of ExperTwin is based on the model of a "Society of Intelligent Agents", where each agent is responsible for a specific task. Collection of JANs from multiple sources, establishing the relevancy, and creation of the personal semantic network are some of the many tasks performed by the individual agents. Tensorflow and Natural Language Processing (NLP) tools have been implemented to let ExperTwin learn from users. Document the design and deployment of ExperTwin as a "Knowledge Advantage Machine" able to search for relevant information while performing a knowledge-based task, is the main goal of the research presented in this paper.**

***Index Terms*—Semantic Web, Knowledge Advantage Machine, Knowledge Base, visualization, Artificial Intelligence, Intelligent Agent**

I. Introduction

Today, the information in every field is expanding exponentially, while simultaneously the half-life of that information is reducing rapidly. Knowledge workers (KW) are facing these twin challenges in keeping themselves current. Traditional methods of continuing education are not sufficient to keep the knowledge workers up-to-date. This situation gives rise to the need for development of an intelligent tool that can automatically gather, refine and present contextually relevant knowledge on demand at the time of need. This paper describes a tool known as ExperTwin that we developed to address these challenges. This may be metaphorically thought of as an alter ego that dwells in cyberspace collecting, refining and transforming information into a usable knowledge base that is always current and relevant. For example, if a knowledge worker, say that a researcher is preparing a research paper, ExperTwin can present the latest relevant annotated bibliography. The same thing applies in any other situation where knowledge plays a key role. Just as Mechanical Advantage [1] played a key role in the industrial era, the concept of Knowledge Advantage could be applied to deal with the information explosion problem [2]. Knowledge Advantage may be simply defined as the ratio of time it takes to accomplish a knowledge based task to amount of time it takes to search for the relevant knowledge. ExperTwin tries to minimize the latter thus contributing to the productivity of knowledge workers. Any machine (or an app) that increases the Knowledge Advantage (KA) may be thought of as a Knowledge Advantage Machine (KAM). ExperTwin described in this paper is an example of a KAM. The major challenges for the effective realization of a KAM are context determination, knowledge organization, knowledge presentation, consistency, and completeness.

A. Context Determination

One of the challenges for a KAM is context awareness. As the KW changes from one environment to another, the information needs may change as well. The capability to follow a human situational context is not present in modern day computers [3]. The presence of context awareness system produces a new learning paradigm with a dynamic context where a KW is able to get knowledge that corresponds with their current context.

B. Knowledge Organization

The KAM system must also have a way for knowledge organization. The information received by KAM is classified according to user context and preferences. Then it is stored in a database. The database stores the ontology which represents the Knowledge Base. This database can also be represented as a Semantic Web that allows data to be shared and reused. Three important technologies for developing the Semantic Web are already in place: 1) JavaScript Object Notation (JSON), 2) Resource Description Framework (RDF), and 3) Graph Database (GDB). A semantic web must have access to structured collections of information and sets of inference rules that researchers can use to conduct automated knowledge search.

C. Knowledge Presentation

Knowledge presentation (KP) in a KAM system is done through a User Interface (UI) with text and graphical capabilities. The KP should present the Knowledge Network stored in the Knowledge Base (KB) in a meaningful way. Some graphical option are 2D, 3D and even Virtual Reality (VR) interfaces.

D. Consistency and Completeness

The ontology used by KAM when creating a Semantic Network needs to be consistent across a community of KAMs sharing their Knowledge Base. Different KAMs can be accessing the same resources which may become redundant if they are not identified as being the same Knowledge Units (KU) referred in this paper as JANs. All KAMs must share a common defined taxonomy of classes of objects and relations among them [4]. A KAM will also require to prove its completeness. The domain of knowledge sources assigned to extract, analyze, and store knowledge to KAM has to be true for every case, where the JAN or KU is always true to source where it was extracted. Therefore, at no time two different KAMs can extract two different KU from the same source. Also two different KAMs cannot extract the same JAN from two different sources.

II. Related work

Many researchers have been involved in the realization of KAM related projects. These are some of the research papers that have been published so far.

A. Vijjana: A Collaborative Agent Model for a KAM

The project named Vijjana [5] envisions an intelligent agent that presents context based information that is drawn from a personalized knowledge base to perform the task at hand more effectively. The paper only discusses the discovery agent, bridging the gap between the presentation agent and the data access layer. It proposes a computer based personal agent that is aware of our personal interests, can determine the context, and offers to present relevant knowledge in an appropriate way. This agent would empower a KW to make better decisions in a shorter time in a collaborative environment. Vijjana is a pragmatic model for collaboratively creating self-organizing domain centric knowledge networks. The discovery agent consists of a browser plug-in, which communicates with the database and performs various operations to process the meta-data information. The Discovery Agent keeps track of the web page details such as the location of the web page, title of the webpage and the description of the web pages along with keywords suggested by the user and saves them in the database for further organization.

B. Empirical Analysis of Workflow Patterns in KAM

The paper published by Dan Sloan et al. [6] presents a work-centric KAM. It studies the workflow pattern or methods by which a user typically utilizes a particular system. A framework is presented by which data mining techniques could be used to extract patterns from an individual work flow data to exploit a type of architecture known as a Knowledge Advantage Machine (KAM). The study empirically demonstrated that a users file system usage patterns can be utilized by a software to automatically and seamlessly learn what is important to the user. The system monitors the file system for usage patterns and makes predictions on personal data usage. Five different classifiers were used to generate training set used to create a predictive algorithm.

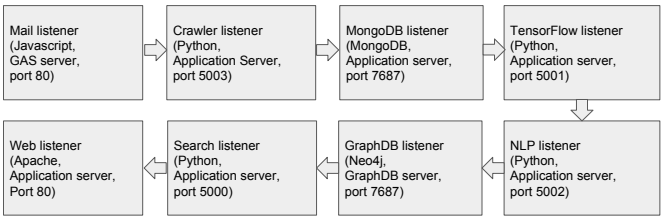

Fig. 1. ExpertTwin Arquitecture.

C. Personal KAM (pKaM): Pervasive Methods for Semantic Web

F. Desjardins et al. [7] present a KAM framework based on the Google Android platform using OpenGL. The pKAM uses a graphical browser to visualize portions of a semantic network. It was composed of the user interface, organizing agent, and database. The input is a set of text files which are processed by an Organizing Agent. The Organizing Agent produces JAN objects and some series of keywords. This is considered a Knowledge Object. The Context Model then locates corresponding taxonomies representing the current user context. Finally, the filtered knowledge objects selected and related references are presented as related knowledge units.

D. User Context Centric Models in a KAM

Luyi Wang et al. [8] proposes a user centric, context and information related taxonomy. In this paper the user profile has two parts. The basic user information and the user preferences where the user preferences are a basic taxonomy. The KAM organizing process classifies the new JANs and finds related JANs. These JANs are organized into taxonomic structures. The reference data repository used was the Open Directory Project (ODP). This is one the largest taxonomy stores for web directories which is organized in hierarchical structure. The KAM ontology contains 14 taxonomies in the first level, 517 taxonomies in the second level and on the third level contains 6056 taxonomies. The JANs are extracted from the user email system, file system, browser bookmarks. It supports three basic types of textual documents: text, pdf, and MS Word

III. Architecture

The KAM ExperTwin project was approached with a n-tier or client-server architecture design in mind. It is composed of the following layers: 1) Presentation layer, 2) service layer (business logic), and 3) data access layer.

A. Presentation Layer

The UI is a Single Page Application (SPA) web site. Web technologies such as HTML5, CSS3, JavaScript, WebGL, JQuery, and AJAX were used for building the tool. User authentication was implemented by using Google Sign in Client API. If a user signs in for the first time, a profile is created in the Graph Database (Neo4j) using the account email as the identifying key. Once the user has been validated, there is a profile settings screen where the user enters the topics

of interest. These topics are stored as keywords in the user profile. The user can add and edit the topics at any time after the validation. Once the user profile is created, the user can choose among the listed topics in the UI. The presentation of the found articles can be either textual or graphical. The textual presentation consists of a list of the articles containing the title, summary, and link to the article. Clicking on an article will open a new tab to the location where the article resides. Another view that can be selected by the user is the graphical view. This view uses the three.js library for 2D/3D interfaces. Once the user selects a topic, the server side service retrieves the matched JANs as a JSON formatted list of the articles via an AJAX call. The retrieved JANs feed the data-driven visualization engine (three.js), this means that it is rendered on every change. There is also a number of ways the articles can be viewed as is shown in Fig. 2. Once an article is clicked, the article is opened on a new tab.

B. Service Layer

The ExpertTwin should extract, process, and present information according to the needs of the KW. The platform chosen is Google Cloud Platform (GCP) running Ubuntu 17. Using a cloud solution freed the project from having to setup server hardware, routers, etc. The programming language chosen for development is Python due to the abundance of Open Source AI libraries. The static site was hosted on an Apache server. and the services built in Python implemented the micro web framework called Flask to be called via REST calls. ExperTwin uses SOA (Service Oriented Architecture) architecture to integrate all the components of the solution. Three physical servers were used (Application, GraphDB and Google Apps servers) with 8 services running. See Figure 1

C. Data Access Layer

The KAM ExperTwin needs to have a way to represent relationships in a knowledge base. This can be described as a Knowledge Graph. This graph contains semantic triples. These are sets of three entities which are subject-predicate object. The final form of a Knowledge Graph is a directed graph or a machine-readable knowledge presentation of the Knowledge Base. The Neo4j Community Edition was selected as the graph database management system for ExperTwin. It has ACID compliant transactional database with native storage and processing. In Neo4j everything is stored in the form of either an edge, a node, or an attribute. Each node and edge can have any number of attributes. Both nodes and edges can be labelled. Labels can be used to narrow searches. The semantic network consists of article has a keyword triplets. The predicate has a contains a weight attribute which is determined by the NLP agent. The article node has the following properties: title, author, link, summary, and rank.

IV. Knowledge Discovery

Our knowledge discovery step for ExperTwin consists of the crawling feature to extract data from various sources. The ExperTwin has the capability to discover knowledge from RSS feeds, email, and documents uploaded via drag and drop on the ExperTwin interface. The web crawler or discovery agent extracts data from the RSS feeds at regular intervals. All the information is inserted in the database as JANs.

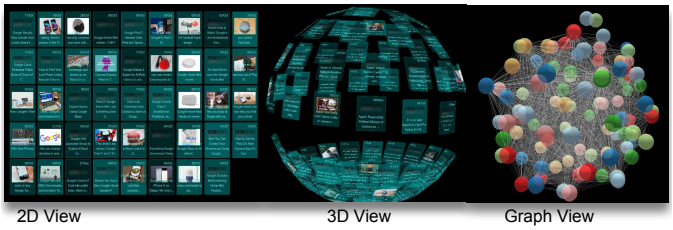

Fig. 2. ExpertTwin 2D View, 3D View (Sphere) and Graph View.

Fig. 3. Virtual Reality View.

A. Input sources

There were 4 different data acquisition mechanisms, a files drag-and-drop area in the user interface, a chrome extension, an email bot, and a Rich Site Summary(RSS) crawler. The drag-and-drop mechanism consists of a special area in the top bar of the user interface that is able to upload multiple files and nested directories. The chrome extension is a button placed in the browser top bar and its function is to send the URL that the user is visiting. The email bot is able to get as input article URLs, RSS URLs and attached files.

B. Web Crawler

For the purpose of this research we selected a Tech News domain. The sources of all the data are close to fifty RSS feeds from various tech news websites. The web crawler or discovery agent extracts data from the RSS feeds at regular intervals. The news feeds are temporarily stored in a Non-SQL MongoDB database until being processed by a TensorFlow model that filters the relevant content as is outlined in section VI (learning agent).

V. Visualization

The applications interface is composed of four main parts: 1) a user setting section, 2) a section of see articles in detail, 3) a 2D/3D reorganization of the articles, and finally 4) a graph view displaying the different relationships between the articles. Additionally, the interface has been built to adapt to any size of device using the open source Bootstrap toolkit. Each section is detailed below. See Figure 4.

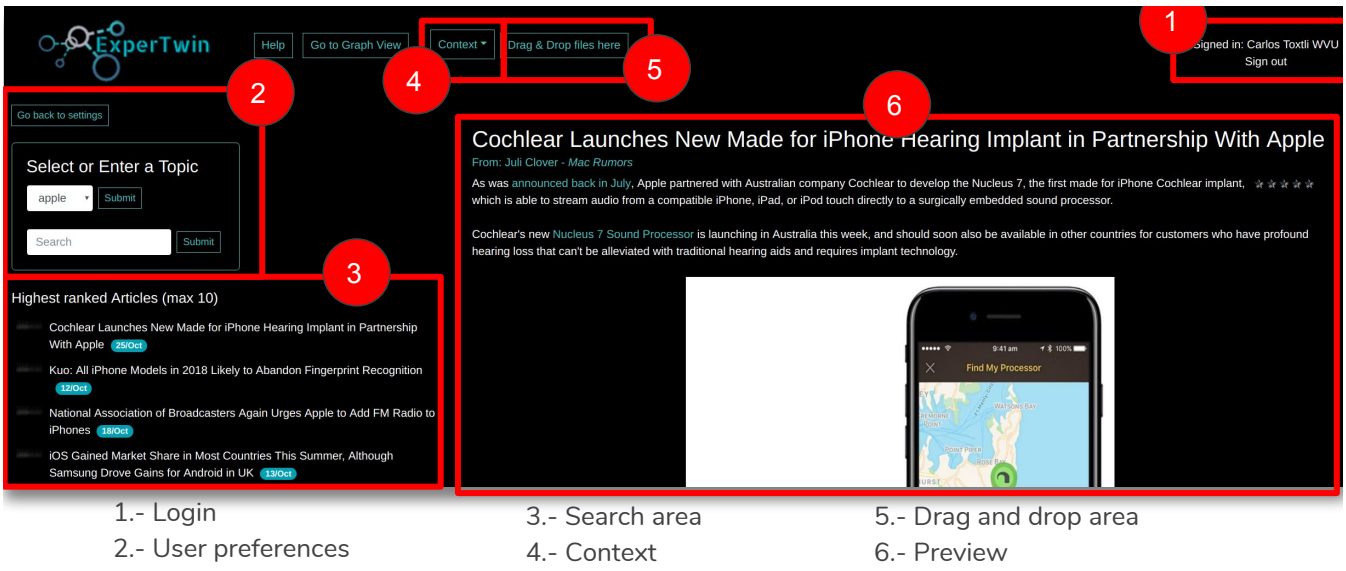

Fig. 4. ExpertTwin Main Interface.

A. User Settings

The first part of the application interface is related to the user settings. Users can have access to their own knowledge-based system. For convenience, since all the backend technologies (server, and databases) were located on the Google Developer Console/Cloud, we decided to integrate Google Sign-In into the application. Once logged in, users can see their topics of interest (keywords), and these define the ranking of articles retrieved by the server queries. At any time, while using the application, the user has the opportunity to change, add and remove any keyword by using the interface Add New and Delete buttons.

B. Article Details View

Once the user has selected a keyword from the dropdown, or has entered a query in the search bar, the second section of the interface will be populated by results from our database. Following the ranking calculation explained in the previous section, a list of maximum 125 appropriate articles will be received by the interface under the form of the JSON file. Of those 125 articles, the top ten, i.e. the ten most highly ranked articles compared to the query/keyword, are displayed as a list (in descending ranking order). We chose to display only ten articles to keep the interface to a reasonable size in height and to avoid a dictionary list effect that goes on and on. If the user, desired to consult more articles, he can still access one of the 125 articles through the 3D view. For each article, a system of ratings is proposed to the user. The rating is composed of five clickable stars, that will light up in yellow or turn back to grey in function of the user choice. The rating will be used by the backend to influence the ranking of the articles.

C. 3D Articles View

In the case where the user wants to see and access more than the ten most highly ranked articles,the user can toggle between the Graph view and the Article View by using the Go to View button located in the navigation part of the interface. There are four types of 3D representations available:Table, Sphere, Helix, Grid additionally to the graph view displaying the different relationships between the articles. See Figure 2.

VI. Learning Agent

ExperTwin learns from the user preferences and interactions to retrieve tailored content to each user through Natural Language Processing and Machine Learning models.

A. Keyword Extraction

Characterization of text articles using keywords has long been researched in open literature [9]. All data processing such as data scraping, organization, filtering, preference learning and data visualization depend on the set of keywords that characterize the JANs therefore it is pertinent to extract the right set of keywords for any given document in the database. Essentially, keyword extraction boils down to using the degrees of words and a different list (i.e. the title of an article) to determine with some degree of accuracy what a passage of text is about. The parameters to this program include the text of the article, the number of keywords desired and optionally the title of the document. Upon receipt of the arguments, Rapid Automatic Keyword Extraction (RAKE) is used to obtain the keywords within the passage of text. Once the RAKE extracted keywords have been handled in the ways previously described, they are then sorted based on their degrees. The higher the degree, the higher the likelihood that the keyword is a good descriptive word of the passage of text parameter. Fortunately, once the words have been sorted, the last step

can be performed, which is constructing the output dictionary. The dictionary is formatted as (Keyword: Weight) with the keywords being the key for each weight. The Keyword portion of the dictionary has already been decided and listed as the top keywords selected by the number of keywords input parameter, but the weight still needs calculated. The weight w_k of keyword k with keyword degree k is calculated as:

$$w_k = \frac{\delta_k}{\sum_{k^k}} \tag{1}$$

Once weights have been calculated and the dictionary constructed, the algorithm will return the dictionary to the programmer

B. Outline of the Learning Agent

This section presents the operational description of the intelligence and adaptability behind the framework discussed so far. Given the collection of all documents referred to as JANs in this framework. It is necessary to extract some features from an initial set of documents based on user defined preferences. The collection of JANs can be thought of a community ontology from which we seek a personal ontology based section. This user defined preferences provide an insight into the ontology of the user therefore within the database and therefore provides a basis for learning component of the overall framework.

C. Feature Extraction

Several methods exist in which terms can be represented in text mining in order to be used as a platform for the learning component. This representation provides a method for evaluating a search heuristic. The most utilized and promising approach has been the vector space model [10]. In this model, a collection of documents D is represented by a m dimensional vector, where each dimension represents a distinct term and m here is the total number of distinct terms used in the collection of documents. Each document in the collection has a corresponding vector representation V, where v_i is the weight of the term d_i for that particular document. Thus the collection of documents is a matrix $D \epsilon R^{nxm}$ where n is the number of documents in the collection. Weights are attached to the relative importance of the words/terms which can be determined using the tf-idf scheme. Using this approach, the term weights are calculated based on their relative importance i.e. how often a term appears in a particular document and how frequently it occurs in the collection of documents D. This is given as

$$w_i = tf_i \cdot log \frac{n}{df_i} \tag{2}$$

Where tf_i is the number of occurrences of term d_i in document D, denoted as the term frequency, df_i here is given as the number of documents which contain the term d_i, this is denoted as the document frequency term. With this measure, the most appropriate words which corresponds to user interests are captured and can then be employed as a set of features for the learning subsystem.

D. Learning Algorithm

The machine learning algorithm employed here is a two-step based method using Artificial Neural Networks (ANNs). The first ANN consists of five layers, one input layer, three hidden layers and one output layer. The second ANN receives classification output from the first and further processes the results for user preferences, this is referred to as the preference learner. This helps monitor user preferences as they change from the defined interest. In order to train the network, an initial set of documents from the database in the server was collected for training purpose. This consisted of 2125 articles from the Web crawler. A set of user defined keywords were generated, this is represented as K(user). Using the keyword extraction scheme as defined above. Keywords are generated from all documents and for document $V_k = 1, 2, ..., n$. Each document class C_k is classified as class 1 or class 0 based on the criteria in equation 3.

$$\mathrm{C}_k = \begin{cases} if^P \left\{ K\left(V_k\right) \cap K\left(User\right) \right\} \geqslant \lambda \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

Here, λ is a threshold which is set as 3 in this article. The classifier was trained using TensorFlow with a relU activation function and using Supported Vector Machine. TensorFlow allows the creation of a predictive model which is used to evaluate every incoming text. For building ExpertTwin, TensorFlow on Python 3.6 has been used to create computational graphs as it promises huge advantage in computational speed. All training data was obtained from the Mongo database in the server. The training data was split into 70:30 training: validation set ratio. A training accuracy of 87% was obtained while a validation accuracy of 85%. With this, the neural network architecture was developed, saved and deployed on the server on the Google cloud computing platform.

E. Preference Learner

As mentioned above, the second ANN incorporates exhibited user preferences for leaning in addition to the first ANN. This preference learner is implemented in the form of neural network with multiple layers that trains upon domain-relevant JANs, some of which have been identified as of primary interest to the user, whereas other JANs in this subset are not the users first preference. The data for training is a subset of the main dataset that was used to collect, discover and catalogue relevant news articles from amongst a set of articles scraped from the Internet. This subset comprises only the articles that have already been classified as JANs in the present domain context. Depending upon the users predisposition towards certain topics within the domain, JANs recording these select topics are rated as one, whereas others are rated as zeroes. The procedure here is to extract all previously

classified documents from the MongoDB database and then vectorize all JANs using the bag of words approach. The dataset is divided into batches. This network consists of three layers with relU activation function. As before, the cross entropy is defined as the objective to be minimized. And the learning algorithm is the back propagation with Adam optimization which adjusts step size in order to find the global minimum of the cost function.

F. Server Based Deployment

Given the learner which is created in an endpoint to be called whenever new JANs are added to the database by the crawler, any article URL can be retrieved and can be parsed using the Article module in the library called newspaper in Python 3. Once the article is parsed and a string of text is returned, the string is transformed based on the master document obtained during the training phase. Thus its feature vector is extracted and passed to the neural network which outputs its probability of being added to the personal ontology database or just remaining in the community database. Furthermore, the corresponding classified JANs are passed to the preference learner for further classification should it match recent users interests.

VII. Discussion

ExperTwin was deployed in a web environment. This setting implies some pros and cons in terms of information acquiring and mobility. The chosen infrastructure was a centralized knowledge base in which the data come from local and remote sources. It is important to highlight the advantages of indexing local files in the same format that remote content is indexed, for instance, in the content creation context, one clear advantage is that we can visualize how related, relevant, and connected the local content is with existing published content. This can benefit authors to better position their content and help them to make it more discoverable. More work is needed to evaluate if this approach derives in a better positioning on search engines.

Interfaces that use 2D graph representation are often hard to analyze when multiple variables are in place and the nodes are densely interconnected. 3D graph interfaces are not by themselves the solution and multiple implications for design should be considered such as the depth of lines, size of nodes, colors, shapes, and even types of interaction. The Virtual Reality version of the ExperTwin visualization was developed to enhance the navigation interaction, see Fig 3. More work in the Mixed Reality and Augmented Reality fields is required to avoid the context switching between the visualization section and the work-space.

VIII. Future Work

ExperTwin is a tool for knowledge workers to get relevant information when needed. Currently the tool helps users open their work-space on every computer although the current version is not mobile friendly. Future work could explore what information workers need from an interface that can be displayed on a mobile phone screen and what kind of activities are more suitable to perform on the go. Other aspect that could be interesting to study is the integration of multiple sensors that are more context aware of the environment. From there the context can be better determined. Syncing mobile device sensors and the desktop client can also extend the experience and can help us to explore how the interface can mutate and behave according with what has been sensed. A proper evaluation of the system should be done to determine if ExperTwin reduces the time that takes to search for relevant information while performing knowledge based tasks.

IX. Conclusion

In this paper we illustrated how a Knowledge Advantage machine could be constructed by providing details of how different requirements could be met. However, we should note the severe limitations of the current implementation of ExperTwin. In future, we hope to develop a plug-and-play feature where a generic ExperTwin could simply be connected to a Knowledge Base. Also, we plan to explore automatic summarization of the knowledge instead of simply presenting the contextually relevant JANs. For example, a researcher engaged in writing papers can be presented with an annotated bibliography relevant paper that is currently being written. Further work on automatic determination of context will be essential for tools like ExperTwin to be successful.

References

[1] Fisher, Len. How to dunk a doughnut: The science of everyday life. Arcade Publishing, 2002.

[2] Hilbert, Martin, and Priscila Lpez. "The worlds technological capacity to store, communicate, and compute information." science 332, no. 6025 (2011): 60-65.

[3] Selviandro, Nungki, Mira Kania Sabariah, and Surya Saputra. "Context awareness system on ubiquitous learning with case based reasoning and nearest neighbor algorithm." In Information and Communication Technology (ICoICT), 2016 4th International Conference on, pp. 1-6. IEEE, 2016.

[4] Makineni, R., Luyi Wang, R. Reddy, and Sumitra Reddy. "Vijjana: A collaborative agent model for creating a knowledge advantage machine." In Collaboration Technologies and Systems (CTS), 2015 International Conference on, pp. 113-117. IEEE, 2015.

[5] Das, Asesh, et al. "Information intelligence in cloud computing: how can Vijjana, a collaborative, self-organizing, domain centric knowledge network model help." Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies. ACM, 2009.

[6] Dan Sloan, Ramana Reddy, Sumitra Reddy, Empirical Analysis of Workflow Patterns in KAM Fourth International Conference on Big Data and Cloud Computing, 2014.

[7] F. Desjardins, R. Moparthi, S. Vangala, L. Wang, Ramana Reddy, Sumitra Reddy, Personal KAM: Pervasive Methods for Semantic Web Morgantown.WV, 2012

[8] Luyi Wag, Ramana Reddy, Sumitra Reddy, and Asesh Das, User Context Centric Models in a KAMOctober 2012, San Francisco, USA.

[9] Y. Matsuo, and M. Ishizuka, Keyword extraction from a single document using word co-occurrence statistical information, International Journal on Artificial Intelligence Tools 13(01), pp.157-169, 2004.

[10] G. Salton, and M. McGill, Introduction to modern information Philadelphia, PA. American Association for Artificial Intelligence retrieval. 1983