A Cognitive Interactive Framework for Multi-Document Summarizer

Anupam Srivastava, Divij Vaidya, Malay Singh, Pranjal Singh and U. S. Tiwary

Abstract In this paper, we present a generic interactive framework based on human cognition, where the system can learn continuously from the Internet and from its interaction with the users. To show the utilization of this framework, Iintelli, an agent based application for multiple text document summarization is developed and compared with the MEAD on the Cran Data Set. Mead is a natural language processing-based summarizer, which provides summary by extracting sentences from a cluster of related documents and Cran is a data set maintained by Information Retrieval Group at University of Glasgow. The human knowledge and experience are represented through fuzzy logic-based word-mesh and sentence-mesh, which can learn. Learning is performed using the competitive models, namely, Maxnet and Mexican Hat Models. As the result shows, the framework performs well as a multi-document summarizer. Though we have tested the framework for multi-document summarization, we believe that it can be extended to develop interactive applications for other domains and tasks.

A. Srivastava (✉) · D. Vaidya · M. Singh · P. Singh · U. S. Tiwary
Indian Institute of Information Technology, Allahabad, UP, India
e-mail: anupam328@gmail.com

D. Vaidya
e-mail: iit2008077@iiita.ac.in

M. Singh
e-mail: iit2008001@iiita.ac.in

P. Singh
e-mail: iit2008008@iiita.ac.in

U. S. Tiwary
e-mail: ust@iiita.ac.in

1 Introduction

We live in a world where the computational capability of machines is increasing at a tremendous rate. However, these machines are still constrained due to lack of human-like intelligence and cognition. Active research in the area of machine learning and Human Computer Interaction has led to the next generation of machines imbibing cognitive capabilities in them. These capabilities represent a significant step towards improving the efficiency and relevance of the results of these systems.

In this paper, we present a generic interactive framework modeled on the human behavior and thinking. The system aims towards blurring the lines of distinction between the data filtered by a human and that done by a machine. The knowledge base used by the system is an integration of data retrieved from the Internet and user knowledge and experience gained through interaction. The prolonged interaction is responsible for dynamic changes in the knowledge base that enables the system to adopt characteristics of its user resulting in understanding the human perception. Human like reasoning and learning abilities have been imbibed in the system using various concepts like competitive learning, fuzzy knowledge representation etc., each represented through a separate agent. All these agents collectively form the framework, which can subsequently be used to develop interactive applications for various domains.

We have found through experiments that use of fuzzy logic is more appropriate to represent human knowledge and experience in our cognitive system. The primary source of information acquisition is the Internet. Moreover, we have used a new form of information gathering method in the task of multi-document summarization i.e. through interaction with the users. This ability to store the episodic memory is realized by the use of Short Term Memory (STM) and Long Term Memory (LTM). This helps in relating to human experiences and memory.

Using cognitive learning techniques the system is able to filter out relevant information from the LTM and Internet, providing its users a personalized information portal, which adapts to their needs. The learning approaches used to replicate human thinking are mainly based on reinforced learning coupled with a lateral inhibitive approach as seen in competitive learning. The storage framework of the system based on LTM and STM, assists in providing the reward/penalty based system, which dynamically modifies the knowledge base and incorporates human-like thinking and reasoning.

The system has been developed using the concepts of self-sustaining software agents. The agents are independent pieces of code that can execute independently. Each agent can be used again in developing an application where there is a need to represent human knowledge and experience. Further, the agents given in this work could easily be clubbed with other new agents to improve upon the information summarization capability of the system.

Various attempts at developing cognitive architecture [1, 2] capable of inference and reasoning on the basis of their knowledge base have been made in the past.

RASCALLI [3–5] is a system developed under a project sponsored by the European Commission. It is capable of performing the above mentioned features. The system tries to model the human metaphor i.e. the system tries to define the mind, various sensors and effectors and the environment from which it continuously learns. The system continuously searches the internet to expand its knowledge base, and works with two forms of memory—LTM and STM.

TextRank [6], developed by Rada Mihalcea and Paul Tarau, is a graph based ranking model for text processing and can be applied for both word abstraction and sentence extraction . In their work, they have used graphs with each node representing a unique lexical unit. The strength of relationship between two nodes is determined by the co-occurrence relation, which is defined in terms of the distance between the two lexical units in the original text. Extending the same principle to sentence extraction, the nodes are used to represent sentences. Here, the similarity between the two sentences was calculated as a function of their content overlap. Content overlap can be found out by use of syntactic filters, or by number of overlapping tokens etc. The sentence similarity was calculated as:

$$Similarity\ (S_i, S_j) = \frac{|\{w_k \in S_i \& w_k \in S_j\}|}{log(|S_i|) + log(|S_j|)} \tag{1}$$

where:

$$S_i = w_i^1, w_i^2, \ldots.., w_i^N. \tag{2}$$

The LexRank [7], developed by Erkan and Radev also performs sentence extraction to create a summary for multiple documents. It is again a graph based method where the nodes represent unique sentences. The level of similarity of two sentences is calculated using cosine similarity of the feature vector formed for the sentences using the inverse frequency—term frequency product.

$$idf\text{-}modified\text{-}cosine(x, y) = \frac{\sum_{w \in x,y} tf_{w,x} tf_{w,y} (idf_w)^2}{\sqrt{\sum_{x_i \in x} (tf_{x_i,x} idf_{x_i})^2} \sqrt{\sum_{y_i \in y} (tf_{y_i,y} idf_{y_i})^2}} \tag{3}$$

Once they have the level of similarity, they calculate the relative importance of a sentence based on the concept of eigenvector centrality in the graph.

$$p(u) = \sum_{v \in adj[u]} \frac{p(v)}{deg(v)} \tag{4}$$

where $p(u)$ denotes the centrality of the node, $adj[u]$ represents the nodes adjacent to node u and $deg(v)$ represent the degree of node v.

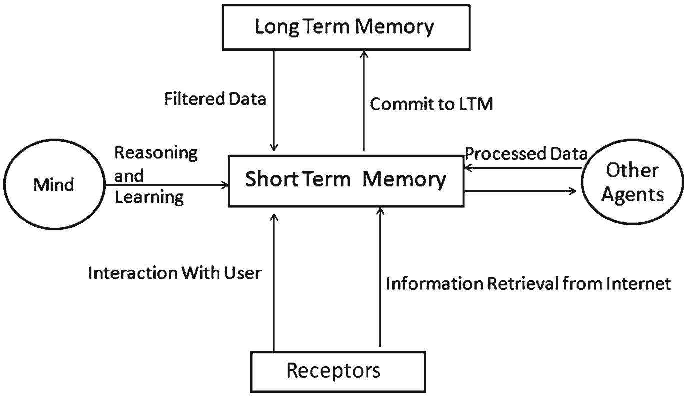

Fig. 1 Cognitive architecture

2 Cognitive Architecture

The proposed architecture models human thinking process. Broadly, the architecture is divided into receptors, agent modules, memory and the mind, which performs reasoning and learning. Each module works in collaboration to produce a result which is similar to the way a human mind would have reasoned and learned to produce the result. Figure 1 shows the interaction between these components.

2.1 Receptors (Sensors)

Receptors are the agents which act as the interface between the external environment and the system. These are the primary source of raw data and essentially contribute towards building and enriching the knowledge base. The agent we have used has its primary source of data as the Internet. Another agent works as an interface between the user and the cognition system. Its primary aim is to interact with the user so that learning can be performed based upon the user inputs and feedbacks. The data gathered by both the receptors is stored in the STM for further processing.

2.2 Memory

The architecture requires both volatile and non-volatile memory to serve different purposes.

The non-volatile form of the memory for permanent storage is called the LTM and the volatile memory is called the STM. LTM serves as the durable centralised database where the information from all the episodes (instances) is stored and updated. It is stored on physical media and is updated through every instance of the application. The data structure used for internal storage is in the form of an undirected graph with nodes and edges. However, the usage of these components varies from one agent to another. The information stored in the LTM models the preferences of its respective user and is used as the knowledge base for perception, reasoning and learning for further episodes of interaction with user.

STM is the temporary work space for the application, where the filtering and processing of data take place. It is directly connected to the various agents which take the raw data, process it and give it to the STM. During each instance, STM is allocated for that particular session, which at the end of the session filters its data and appropriate commits are made to the LTM which includes forming a perception like the user.

2.3 Application Specific Agents

Once we have the raw data acquired by the receptors and stored in the STM, specific agents are invoked to process and filter out the data in user understandable form. Due to the agent based programming methodology of the framework, the number of such agents can be varied as per the needs of the application. In this work, we use a number of agents for tasks like, word abstraction, sentence extraction, etc. for text summarizing, which collaborate with each other and mind agent in carrying out the cognitive processes. At the moment, the agents are designed to perform the task of text summarization. However, the same architecture can be used for other domains and tasks, by changing the agents at this level.

2.4 Mind

This agent is responsible for carrying out the reasoning and learning process in order to generate the summary. Moreover, it tries to form a perception about the user and its environment in order to give more accurate results. Learning and perception are carried out by manipulation of fuzzy parameters, which essentially change the priority of data in the LTM with respect to a particular user. The user feedback and actions are utilized to perform reinforced learning. Depending upon whether the user likes or dislikes the produced summary, the fuzzy edge weights are changed in order to map the system model to the user mind model.

We maintain a separate log for each user so that, perceptions for individual users can be created through manipulations of edge weights. For example, depending upon previous user actions, the system can understand that the user perceives apple as a fruit rather than Electronics company. Reasoning is performed by spreading activations in the word-mesh and the sentence-mesh. Relations among nodes which are not directly connected are realized on the fly, depending upon the user input and environmental conditions.

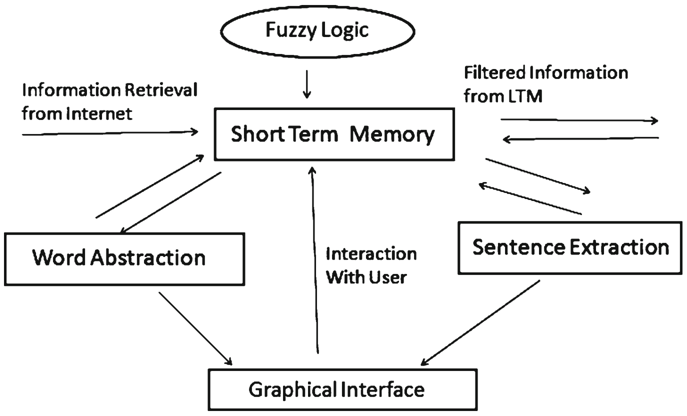

Fig. 2 Interaction between different agents in Iintelli

3 The Proposed Framework: Iintelli

To show the utilization of our cognition system, we have developed an agent based application, Iintelli, which is capable of multiple document text summarizations. The application consists of various agents, which are independent and self-sustaining in their respective fields of work. The system has two different agents to perform text summarization, one for word abstraction and another for sentence extraction. Other agents interact with the user and represent human knowledge and experience using competitive learning models, namely Maxnet and Mexican Hat. While interacting with the user, just like a human, the system tries to develop a perception about the current environment. This is the most important feature of our text summarizer since it allows the user to connect and express more accurately, and hence, produce coherent results.

Figure 2 shows the interaction between the different modules. The agent for information acquisition sends the raw data to the STM where the mind performs reasoning and learning depending upon the current perception, and passes on the filtered data to the word abstraction and sentence extraction agents. These agents perform summarization as described below and present the result to the user with the help of a GUI. The user may then perform actions and give feedback, which is then used for reinforced learning by the system.

3.1 Learning System Through Fuzzy Logic-Based Word-Mesh and Sentence-Mesh

The design followed in the system performs learning and reasoning by manipulating a set of parameters and weights, which determine the correlation of different textual units of the text. To represent these parameters, we have used fuzzy values with sigmoid membership function. There are many other membership functions, which could have been used like triangular membership function, trapezoidal membership function, etc. However, after several trials, it was found that the sigmoid membership function [8] produces the best result. The function is defined as

$$p_x^{\lambda}(x) = \frac{1}{1 + e^{(-\lambda(x-a))}} \tag{5}$$

3.2 *Word Abstraction*

This agent assigns high importance of to the keyword provided by the user and works on its derivatives and related keywords. The central idea to word abstraction is picking up common features among words and using these features to assign a weight to each of the related words. In the context of Iintelli, the keywords provided by the user are given the maximum weight and word abstraction is employed on the source document. This yields words with varying weights to be introduced to the knowledge base.

Basic Structure of Long Term Memory (Multi-dimensional Fuzzy Word Mesh)

The main data structure for the long-term memory is an undirected graph. The nodes of the graphs are words, which have been found after using word abstraction technique in many iterations of the system. The edges of the graph are weighted edges signifying the intensity of relationship between the connected words. Sometimes, the relationship between the words may vary according to the context; hence, the edges also store the name of the topic in which context the weight is assigned. Thus, two nodes may have multiple edges between them. The importance of semantic relationship among words is not gauged by absolute crisp numbers; instead, a fuzzy value is assigned to it based on the information gained from the training data. The fuzziness ensures the absence of crisp logic, which sometimes may render false negatives.

Knowledge Base Filtering

To prepare the word mesh, the source document is parsed through once and the sentences containing the original key words are picked up for further processing. Further processing would include word abstraction i.e. finding out syntactic and semantic relationship [9] among words in sentences and storing the words related to the given keywords in the database.

Once the word mesh has been prepared with all its associated weights in terms of fuzzy values, the whole document is passed again to include those sentences also into consideration that have any of the words in the mesh present in them. To do this each word in the sentence is checked for its presence in the word mesh and if it is found, then according to its edge in the word mesh, its weight is added to the overall score of the sentence. A point to note in this context is that the overall score is also computed as a fuzzy value. Once we have the sentences with each of them having its own "score" in the form of a fuzzy value, the sentences are put in a priority queue whose implementation is modified to give the best sentence in terms of score. The overall score is computed as

$$Score = \sum_{j=0}^{k} weight_j \tag{6}$$

k is the number of words of the database present in the sentence.
$weight_j$ is the fuzzy weight of the $word_j$.

Reinforcement Using Maxnet

The approach towards text summarization uses reinforced learning. From the interactive user interface, the user either accepts a sentence as valid or penalizes a sentence. The penalty imposed on the sentence is updated throughout the word mesh through successive activations. All the words that are present in the sentence and are a part of the word mesh are penalized by a constant factor in which its crossover point is reduced.

The user interaction leads to reinforced learning, which also includes competitive learning through the use of winner takes all strategy. This form of lateral inhibition is like that of Maxnet where the winning node is awarded whereas all the nodes, which are penalized by the user, have their weight reduced by a finite amount.

3.3 Sentence Extraction

The agent is used for extracting important and useful sentences directly from the documents based upon some predetermined features. This is a very useful technique for performing text summarization of multiple documents.

The algorithm works by creating a mesh where each node represents a sentence. The edges between two sentences represent the level of similarity of the two sentences. The level of similarity is calculated by making a feature vector for each sentence. Once, we have the feature vector for both the sentences, we calculate the similarity between these two sentences which is represented as edge weights between two given nodes. To reduce the computational load, we put use a threshold

T1 to remove those edges, which have weights less than this; they would be weak and hence, would not contribute to the result significantly.

After this, we extract those sentences, which seem to be important to all the documents. This is done by using a form of competitive learning to let the weighted graph converge, so that we can extract the top ranked sentences. Unlike the previous agent, we use Mexican Hat for performing competitive learning, which is basically the soft winner takes all approach of competitive learning. The basic idea is that, the nodes which would have a strong centrality factor associated with them would increase the rank of those nodes which share a strongly weighted connection with them. A threshold T2 is used to determine the number of epochs after which one could say that graph has converged. However, to avoid pre mature convergence of the graph and noise reduction, we add randomness in the algorithm. This randomness ensures that it does not get stuck in local minima of the state space. Rather, it allows one to walk around randomly, giving incentives to the node (and its strongly associated nodes) where it is present currently until the graph converges.

Once the graph converges, we pick the sentence with the highest centrality values and present it as the summary of the documents in question.

4 Experiments

The implementation has been done using C++ and Qt Library. To demonstrate the utility of the application, we have performed experiments on the Cran Data Set [10], which comprises of 1400 text documents from various fields and compared it with the results produced by MEAD [11] which is a natural language processing-based summarizer. We divided the data set into groups of ten documents each and used both the agents (sentence extraction and word abstraction) for summarizing. The summaries generated were then compared to calculate the percentage match. The percentage match was calculated as the ratio of the total number of matched sentences and the total number of sentences in the summary for each data set. We compared the performance of the system against MEAD, with and without the fuzzy component.

Figure 3a shows the comparison between MEAD and Iintelli without the fuzzy component. The graph shows how the output is affected as we vary the threshold, which determines the convergence of the graph. Figure 4a shows the comparison between MEAD and Iintelli with the fuzzy system component. The graph shows how the output gets affected as we vary the initial weights, which determine the relationship strength among different textual units and with the slope of sigmoid fuzzy numbers.

Figures 4b, 5a, b show how the percentage match varies among the different document sets. We have plotted bar graphs for only those constant values which give the best result in each corresponding experiment. Figure 4b shows the distribution of percentage match among the document sets with $Slope = 0.7$ and varying NN weight. Figure 5a is plotted with $T2 = 2 \times 10^{-4}$ and varying $T1$ whereas Fig. 5b is plotted with $Slope = 0.7$ and varying $T1$.

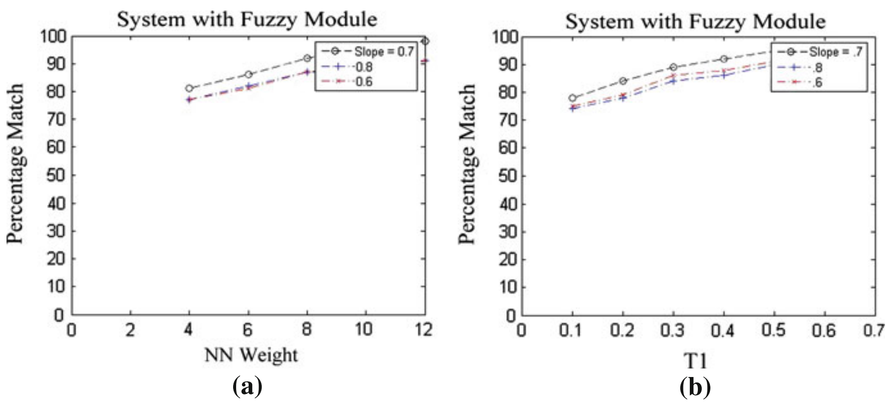

Fig. 3 Comparison between MEAD and Iintelli with fuzzy component. **a** Varying NN weight, **b** varying T1

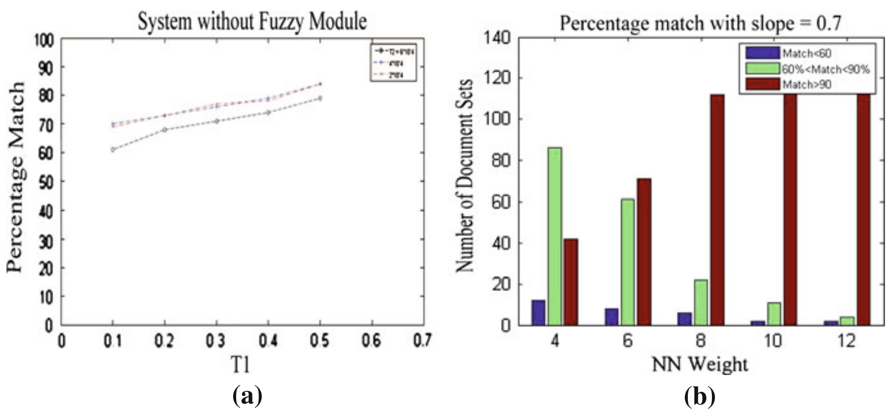

Fig. 4 **a** The comparison between MEAD and Iintelli without the fuzzy component. **b** Distribution of percentage match among the document sets with $Slope = 0.7$ and varying NN weight

Clearly, one can observe that Iintelli works better with the fuzzy component as we are able to represent human knowledge more appropriately. Also, it can be seen that the competitive models are able to output, almost all the sentences, which the MEAD does. Moreover, due to interaction with the user, the system is able to produce more coherent results. The system appears to work the best with the slope of fuzzy number set to 0.7. Moreover, we observe that as we increase the threshold T1, the results improve and eventually these saturate beyond 0.4. The system seems to work best with T2 set to 2×10^{-4}. The bar graphs shows that with $Slope = 0.7$ and $T1 = 0.5$, we are able to get greater than 90 % in over 120 document sets out of 140.

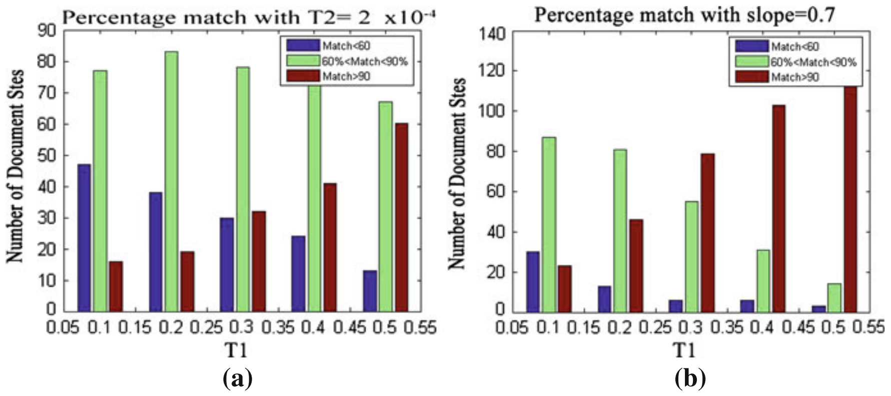

Fig. 5 Distribution of percentage match among the document sets with varying T1 **a** $T2 = 2 \times 10^{-4}$ **b** Slope = 0.7

5 Conclusion and Future Work

We have presented hybrid cognitive system which is capable of multi-document summarization through competitive learning models. We have developed an application, Iintelli, a multiple document text summarizer, which is capable of learning using competitive learning models and compared it with MEAD. The results of these experiments show that the system is able to produce similar output as MEAD. Moreover, the text summarizer allows human computer interaction, which obviously gives out better and more coherent results. The experiments also show that we get better results, if fuzzy logic is used for representation of human knowledge and experience.

To show that the framework works well with other applications also, the same system can be extended for image ranking and association with the help of different set of features. One can also develop an abstractive summarizer or other knowledge discovery and generation applications.

References

1. Skowron, M., Irran, J., Krenn, B.: Computational framework for and the realization of cognitive agents providing intelligent assistance capabilities. In: 18th European Conference on Artificial Intelligence, Cognitive Robotics Workshop, pp. 88–96 (2008)
2. Xu, F., Adolphs, P., Uszkoreit, H., Cheng, X., Li, H.: Gossip Galore: A Conversational Web Agent for Collecting and Sharing Pop Trivia. In: Filipe, J., Fred, A., Sharp, B. (eds.) Proceedings of ICAART 2009—First International Conference on Agents and Artificial Intelligence. 19–21 Jan 2009. Porto, Portugal (2009) (Forthcoming)
3. Eis, Ch.: RASCALLI platform: a dynamic modular runtime environment for agent modeling. Master's Thesis, Vienna University of Technology, Vienna, Austria (2008)

4. Kostadinov, S., Grinberg, M.: The embodiment of a DUAL/AMBR based cognitive model in the RASCALLI multi-agent platform. In: Proceedings of 8th International Conference on Intelligent Virtual Agents (IVA-08). LNCS, vol. 5208, pp. 356–363 (2008)
5. Eis, Ch., Skowron, M., Krenn, B.: Virtual agent modeling in the RASCALLI platform. In: PerMIS'08— Performance Metrics for Intelligent Systems Workshop, Gaithersburg, MD, USA, August 19–21 (2008)
6. Mihalcea, R., Tarau, P.: TextRank: bringing order into texts. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona, Spain, July (2004)
7. Radev, D.R., Erkan, G.: Lexrank: graph-based lexical centrality as salience in text summarization. J. Artif. Intell. Res. (JAIR) **22**, 457–479 (2004)
8. Dombi, J., Gyorbiro, N.: Addition of sigmoid-shaped fuzzy intervals using the Dombi operator and infinite sum theorems, Fuzzy Sets Syst. **157**(7), 952–963 (2006)
9. Li, Y., McLean, D., Bandar, Z.A., O'Shea, J.D., Crockett, K.: Sentence similarity based on semantic nets and corpus statistics. Knowl. Data Eng. IEEE Trans. Data Eng. **18**(8), 1138–1150 (2006)
10. Glasgow IDOM—Cranfield collection: http://ir.dcs.gla.ac.uk/resources/test_collections/cran/. Accessed 14 Apr 2011
11. MEAD: http://www.ncibi.org/gateway/mead.html. Accessed 14 Mar 2011