Abstract: Nowadays, the need for intelligent software systems as personal Meta Search Engines which are capable of supplying user by needed information from massive information resources is sensible. Moreover, the current tools have many deficiencies. In a Meta Search Engine, proper queries based upon user's interests are sent to different search engines (or in general, information servers). Then, the returned results are refined and made available for the user based on their priorities. On the same direction in this study we try to design, implement and examine a complete architecture for a customized software intelligent agent which is able to retrieve information from multiple sources based on user interests. The designed architecture is able to use information fusion methods to achieve this goal. Here, the OWA, which is a member of the fuzzy integral operators family, has been used. This agent performs fusion in data, feature, and decision levels. Also the agent is able to extract the behavioral model of each information server against various subjective clusters to increase the intelligence of searching methods and obtain higher quality results in searching process.

Keywords: data/information fusion, list fusion, Meta search engine, intelligent agent, and subjective cluster.

1 Introduction

Undoubtedly, high performance information retrieval from internet and other large and very large scale data sources is one of the most important problems in efficient use of information sources. Nowadays, web is the largest data source of documents and other forms of information and a suitable ground for evaluating the different Information retrieval techniques. The more the web is expanded, more the need for powerful search tools become evident. At the present time, there are lots of services for web search, but none of them are helpful as expected and actually in the most cases the results are unsatisfactory.

Up to now, some researches using information fusion techniques for information retrieval purposes are done. But none of Meta Search Engines used OWA\(^\text{1}\) for fusing results in a way that we explain in this paper. Beside it, the problem of behavioral model extraction of information servers against different subjective clusters is used for the first time in this paper from this point of view.

2 The Characteristics of Information Fusion Based Intelligent Meta Search Engine

In this paper, an intelligent Meta Search Engine is discussed which uses information fusion techniques. This agent plays the role of a customized Meta Search Engine for the user. This agent receives the words or phrases interesting for the user who is willing to find the subjects related to them, and then it asks the user to determine the weights of importance of words, whether or not to be used in the text. Weight has a linguistic concept here. It means that the user can determine the importance of whether or not word must be used in the text as Low, Medium, High, and Very High. Then the agent makes ready the queries by "Query Generator" unit according to the number of information servers (e.g. internet search engines, and/or data sources).

After sending queries, each server returns a list of ranked documents based on their proximity to the subject, and the algorithm upon which the server works. Then the Meta Search Engine reviews these lists and then eliminates the repeated items and fuses

\(^\text{1}\) - Ordered Weighted Averaging operator
them based on the list fusion algorithms in a way that a ranked list of documents is prepared. In this list a score is given to each document based on its position in the list [1].

Then the documents are processed one by one in this list, and their conditions are determined regarding whether or not the keywords or phrases selected by the user to be present and according to their presence quantity and distribution, two scores are given to each document. In this mechanism, Ordered Weighted Averaging operator (OWA) [2,3] has the basic role. In this paper we will not focus on the fusing method of the lists attained from search engines and how the Meta Search Engine makes a model from search engines used by it. Each time the user decides to use the Meta Search Engine, he or she specifies that this interesting subject is in which subjective cluster [4].

By **subjective cluster**, we mean "a logical classification of interesting subjects". Each time the user starts a new search he or she can select from the available clusters or create a new cluster. Some of available clusters at the present are shown in table 1.

### Table 1: Subjective clusters which the agent holds their information to identify behavioral model of information servers.

<table>
<thead>
<tr>
<th>No.</th>
<th>Cluster Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data/Information Fusion</td>
</tr>
<tr>
<td>2</td>
<td>Context sensitive web searching</td>
</tr>
<tr>
<td>3</td>
<td>Dempster Shafer theory</td>
</tr>
<tr>
<td>4</td>
<td>Fuzzy Controllers</td>
</tr>
<tr>
<td>5</td>
<td>CORBA</td>
</tr>
<tr>
<td>6</td>
<td>TBM² Model</td>
</tr>
<tr>
<td>7</td>
<td>Neuro Fuzzy Systems</td>
</tr>
</tbody>
</table>

The agent has allocated one score to each server based on its previous performance in each subjective cluster. The agent allocates the score of the server which has retrieved the document to it. These scores become updated after each use of Meta Search Engine, based on an algorithm which we will not discuss about its details in this paper. So gradually the behavioral model of each search engine and its efficiency on a special subject is formed in the mind of Intelligent Meta Search Engine. Regarding the mentioned explanations, 4 different scores are gained for each document that at the end the agent must calculate the final score of each document based on them and represent it to the user. It is done by fusing of these scores by information fusion methods and for each score a weight is considered which shows the importance of that criterion at the final decision. Finally, the score of each server in the special subjective cluster is updated with / without feedback received from the user. In this paper we will explain the methods that implement the above explanations.

### 3 Information fusion using Ordered Weighted Averaging (OWA) operator

In Ref. [3] Yager introduced the OWA operator. We recall that:

\[
OWA\left(a_1,\ldots,a_n\right) = \sum_{j=1}^{n} w_j b_j \quad \text{where} \quad b_j = \text{the } j\text{-th largest of the } a_i \quad \text{and} \quad w_j \text{ are a collection of weights where} \quad w_j \in [0,1] \quad \text{and} \quad \sum_{j=1}^{n} w_j = 1. \quad \text{Thus we can show OWA as a inner product like} \quad OWA_w\left(a_1,\ldots,a_n\right) = W^T B.
\]

In this paper if we let \( \text{index} \left(j\right) \) be the index of the \( j\)th largest of the arguments \( a_i \) then we can express

\[
OWA\left(a_1,\ldots,a_n\right) = \sum_{j=1}^{n} w_j a_{\text{index}(j)} \quad (1)
\]

Collectively he denoted a \( n \) vector \( W \) whose components are the \( w_j \) as the OWA weighting vector. We shall say that a vector having the properties

\[
w_j \in [0,1] \quad \text{and} \quad \sum_{j=1}^{n} w_j = 1 \quad \text{is a proper vector. By appropriately selecting} \quad W \quad \text{we can obtain the preceding valuations. For} \quad W = W^w \quad \text{where} \quad w_1 = 1 \quad \text{and} \quad w_j = 0 \quad \text{for} \quad j \neq 1 \quad \text{we get the optimistic valuation. For} \quad W = W^o \quad \text{where} \quad w_n = 1 \quad \text{and} \quad w_j = 0 \quad \text{for} \quad j \neq n \quad \text{we get the pessimistic valuation. For} \quad w_j = \frac{1}{n} \quad \text{we get the neutral.}
\]

Yager has shown that OWA is able to act as a soft computing tool and mimic an expert. It is proved that OWA is a member of Choquet fuzzy integral operator family [5,6].

The OWA operator and related aggregation techniques such as the Choquet and Sugeno integrals provide methods for formulating multi-criteria decision functions.

OWA as a powerful tool has been used in a variety of applications such as decision making, expert systems, text learning, fuzzy control, and so on[7,8]. The OWA mechanism consists of two main parts. The inner product is linear part and the sorting process of arguments is its nonlinear part. It will be shown that how a proper RIM² function, could be found using fuzzy rules, to calculate OWA weights.

### 4 The information fusion intelligent agent's architecture

In this section we explain the main part of designed architecture. First we define the n-gram concept. A **n-gram** is a sub-sequence of \( n \) items from a given

---

2. Common Object Request Broker Architecture
3. Transferable Belief Model
4. Regular Increasing Monotone
sequence. n-grams are used in various areas of statistical natural language processing and genetic sequence analysis. The items in question can be letters, words or base pairs according to the application. An n-gram of size 1 is a "unigram", size 2 is a "bigram", size 3 is a "trigram", and size 4 or more is simply called an "n-gram" or "(n-1)-order Markov model" [6, 9, 10].

n-grams find use in several areas of computer science, computational linguistics, and applied mathematics.

4.1 Calculating the associated score to documents based on number of interested n-grams

Number of interested n-grams and how they are distributed is one of the most important parameters in scoring documents. For example the repetition frequency of an n-gram in a document has a strong correlation with the degree of relation between the document and that n-gram. On the other hand in many cases user doesn’t like special n-grams exist in the desired document. Thus for more refinement it is necessary that the agent consider this constraint and associate the negative score to the presence of those n-grams. In implementation of this agent we considered not more than trigrams. User can assign a weight to presence or absence of an n-gram from the following set:

\{Low, Medium, High, VeryHigh\}

Figure 1 illustrates a part of agent’s architecture. User interface gets user interests and shows results to him/her.

Now we are going to explain the algorithm. First it is needed to define required parameters.

\(N\) : Overall number of documents
\(T\) : Number of interested n-grams
\(p_{kj}\) : Number of \(j^{th}\) interested n-gram in \(k^{th}\) document
\(\alpha_j\) : Importance degree of presence (absence) of \(j^{th}\) n-gram
\(\epsilon_{kj}\) : The score assigned to \(k^{th}\) document related to \(j^{th}\) interested n-gram
\(\theta_{kj}\) : Normalized form of \(\epsilon_{kj}\) \(0 \leq \theta_{kj} \leq 1\)

In all of the above definitions we have:

\(j = 1, \ldots, T\) and \(k = 1, \ldots, N\)

One proper value for \(\epsilon_{kj}\) is \(p_{kj}\) and then from the definition of \(\theta_{kj}\) we have:

\[
\theta_{kj} = \frac{\epsilon_{kj}}{\max_{l=1}^{N} \epsilon_{lj}}
\]

(2)

So, \(T\) scores are obtained for each document. In next step we combine this scores such that one score is associated to each document based on its number of interested n-grams. To achieve this goal the OWA operator is used. The final decision function using this operator is:

\[
\xi_{kj} = \alpha_j \theta_{kj}
\]

(3)

\(\xi_{kj}\) is the index of the \(j^{th}\) largest \(\xi_{kj}\).

In this process the quantifier is used to calculating weights of OWA [2]. A useful approach to the determination of the OWA weights makes use of a class of functions which is called Basic Unit interval Monotonic, BUM, functions. A BUM function is a mapping \(f : [0,1] \rightarrow [0,1]\) such that \(f(0) = 0\).
\( f(1) = 1 \) and \( f(x) > f(y) \) if \( x > y \). Using these functions we can obtain a set of proper weights as:

\[
w_j = Q \left( \frac{j}{T} \right) - Q \left( \frac{j-1}{T} \right), \quad j = 1, \ldots, T . \quad (4)
\]

It means that: \( w_1 = \left( \frac{1}{T} \right)^a, \ldots, \)

\[
w_j = \left( \frac{j}{T} \right)^a - \left( \frac{j-1}{T} \right)^a, \ldots, w_T = 1 - \left( \frac{T-1}{T} \right)^a
\]

Simply it can be shown that \( \sum_{j=1}^{T} w_j = 1 \). Now with two simple fuzzy rules we can find the appropriate range for \( a \) [11].

**First Rule:** If more than \( m \) percent of allocated scores to a document are greater than \( y \), then final score associated to that document must be greater than \( x \).

**Second Rule:** If more than \( m' \) percent of allocated scores to a document are less than \( y' \), then final score associated to that document must be less than \( x' \).

In above rules: \( x, y, x', y' \leq 1; m, m' \leq 100 \). Then for the first rule (in drastic boundary form) we have:

\[
\left( w_1 + \ldots + w_{\left[ \frac{m}{100} \right]} \right) y + \left( w_{\left[ \frac{m}{100} \right]} + \ldots + w_T \right) x \geq 0 \geq x
\]

which leads us to upper bound for \( a \), and on the other hand for the second rule we have (in a drastic manner):

\[
\left( w_1 + \ldots + w_{\left[ \frac{m'}{100} \right]} \right) y + \left( w_{\left[ \frac{m'}{100} \right]} + \ldots + w_T \right) x' \leq x'
\]

which leads us to lower bound for \( a \). With some simple calculations it is obtained that:

\[
\ln \left( \frac{x - y'}{1 - y'} \right) \leq a \leq \ln \left( \frac{m}{y} \right) \quad (5)
\]

It can be seen that above relation implies that \( x < y \) and also \( x' < y' \).

The following example shows the process of calculating weights.

**Example 1:** Suppose that from an expert point of view \( m = 80, y = 0.8, x = 0.65 \) for first rule and \( m' = 60, y' = 0.4, x' = 0.75 \) for the second, might be suitable values. Also suppose that 5 n-grams \( (T = 5) \) are specified as interested keywords and now the 7th document from ranked list is being processed. The needed information is illustrated in table 2. Then if we choose \( a \) from [0.5882, 0.9305], the both rules will be satisfied. With \( a = 0.75 \), which converts OWA to an Or-like (Optimistic) operator and calculating weights, \( U_j \) will be obtained:

\[
U_j = \sum_{i=1}^{n} w_j \xi_{ij} = 0.5731
\]

Orness characterizes the degree to which the aggregation is like an OR (Max) operation and is defined as the following definition [12]:

\[
Orness(W) \equiv \frac{1}{n} \sum_{i=1}^{n} (n-i) \xi_{ij} \quad (6)
\]

### 4.2 Calculating the associated score to documents based on distribution of interested n-grams

Number of n-grams is not a perfect factor to deciding about the relevance of a document with the user’s interest. The distribution of n-grams in a document and also the volume of it are very important factors. This is a complex problem. For example suppose that the number of a specific n-gram in a document with 1000 words is equal by the number of that n-gram in a document with 2000 words.

<table>
<thead>
<tr>
<th>( j )</th>
<th>( j )th n-gram</th>
<th>( \alpha_j )</th>
<th>( p_{ij} )</th>
<th>( \varepsilon_{ij} = p_{ij} )</th>
<th>( \frac{N}{\sum_{i=1}^{N}} \left( \varepsilon_{ij} \right) )</th>
<th>( \theta_j )</th>
<th>( \alpha_j \theta_j )</th>
<th>( \xi_{ij} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CORBA</td>
<td>Very High</td>
<td>84</td>
<td>84</td>
<td>84</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>JAVA RMI</td>
<td>Very High</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>0.3333</td>
<td>0.3333</td>
<td>0.5100</td>
</tr>
<tr>
<td>3</td>
<td>BANK SERVER</td>
<td>Medium</td>
<td>47</td>
<td>47</td>
<td>56</td>
<td>0.6800</td>
<td>0.6800</td>
<td>0.5100</td>
</tr>
<tr>
<td>4</td>
<td>IDL</td>
<td>High</td>
<td>17</td>
<td>17</td>
<td>25</td>
<td>0.3485</td>
<td>0.3485</td>
<td>0.2614</td>
</tr>
<tr>
<td>5</td>
<td>ORB</td>
<td>High</td>
<td>23</td>
<td>23</td>
<td>66</td>
<td>0.2614</td>
<td>0.2614</td>
<td>0.2614</td>
</tr>
</tbody>
</table>

To gain a model using this criterion again the required parameters will be defined.

The definitions of \( N, T, p_j \) and \( \alpha_j \) are as their definitions in 4.1 section.
\(d_{ikj}\): Position of \(i^{th}\) occurrence of \(j^{th}\) n-gram in \(k^{th}\) document

\(n_k\): Overall number of words in \(k^{th}\) document

\(D_{kj}\): Set of \(j^{th}\) interested n-gram's occurrence positions in \(k^{th}\) document. It means:

\[D_{kj} = \{d_{ikj} \mid i = 1, \ldots, p_{kj}\}\]

To have a proper metric to comparison we define:

\[n' = \max_{i=1}^{N} \left(n_r\right)\]

and then

\[d'_{ikj} = \frac{n'}{n_k} \text{ and } D'_{kj} = \{d'_{ikj} \mid i = 1, \ldots, p_{kj}\}\]

Then we can define:

\[\gamma_{kj} = \sum_{d_{ikj} \in D_{kj}} (n' - d'_{ikj}) \text{ and normalized form of it is as following:} \]

\[\eta_{kj} = \frac{\gamma_{kj}}{\max_{i=1}^{N} \left(\gamma_{ij}\right)} = \frac{\sum_{d_{ikj} \in D_{kj}} (n' - d'_{ikj})}{\max_{i=1}^{N} \left(\gamma_{ij}\right)} \] (7)

In all of the above definitions we have:

\(i = 1, \ldots, p_{kj}\) and \(j = 1, \ldots, T\) and \(k = 1, \ldots, N\)

Thus for \(k^{th}\) document, \(T\) scores are obtained and as the previous case, we must fuse them to calculate a single score for each document. For this purpose the OWA with the weight vector \(W = [w_1, \ldots, w_T]^T\).

So

\[P_k = \sum_{j=1}^{T} w_j \alpha_{kj}; r = \text{index} \left( j \right) \]

\[\alpha_{kj} = \alpha_j \eta_{kj} \] (8)

Here with a same process as the previous section, the quantifier \([Q_j(r) = r^a] \geq 0\) is used to determination of

Table 3: Information related to distribution of interested n-grams in a document related to CORBA technology

<table>
<thead>
<tr>
<th>(j)</th>
<th>(j^{th}) n-gram</th>
<th>(\alpha_j)</th>
<th>(p_{kj})</th>
<th>(\gamma_{ij})</th>
<th>(\eta_{kj})</th>
<th>(\alpha_j \eta_{kj})</th>
<th>(\omega_{kj})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CORBA</td>
<td>Very High</td>
<td>84</td>
<td>641512</td>
<td>641512</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>JAVA RMI</td>
<td>Very High</td>
<td>2</td>
<td>19429</td>
<td>123218</td>
<td>0.1577</td>
<td>0.1577</td>
</tr>
<tr>
<td>3</td>
<td>BANK SERVER</td>
<td>Medium</td>
<td>47</td>
<td>205304</td>
<td>218300</td>
<td>0.9405</td>
<td>0.4702</td>
</tr>
<tr>
<td>4</td>
<td>IDL</td>
<td>High</td>
<td>17</td>
<td>133577</td>
<td>244817</td>
<td>0.5456</td>
<td>0.4092</td>
</tr>
<tr>
<td>5</td>
<td>ORB</td>
<td>High</td>
<td>23</td>
<td>114183</td>
<td>269851</td>
<td>0.4231</td>
<td>0.3173</td>
</tr>
</tbody>
</table>

OWA weights. The following example shows the process:

**Example:** In the previous example suppose that \(n_\gamma = 8781\) and \(n' = 10021\). The complementary information about the process is shown in table 3. With the same rules that we used in previous example and \(\alpha = 0.75\) the weight vector is

\[W = [0.2991, 0.2039, 0.1788, 0.1642, 0.1541]^T\]

and \(P_k = \sum_{j=1}^{5} w_j \alpha_{kj} = 0.5445\). This value will be used as the associated score of \(7^{th}\) document based on distribution of 5 n-grams in the text.

### 4.3 Final Score Calculation

As it was explained, four scores are calculated for each document. Because each of these scores has different importance in the final decision making, we assign an importance coefficient \(\beta_i\) to each of them \(\left(\sum_{i=1}^{4} \beta_i = 1\right)\). These scores are:

- The score which is obtained from list fusion mechanism shown in Figure 2 (the mechanism is not mentioned in this paper) for the \(k^{th}\) document: \(V_k\) with importance \(\beta_1\).
- The score related to number of interested n-grams in the \(k^{th}\) document: \(U_k\) with importance \(\beta_2\).
- The score associated to distribution of interested n-grams in the \(k^{th}\) document: \(P_k\) with importance \(\beta_3\).
- The score associated to the server which has retrieved the \(k^{th}\) document (The algorithm of updating this score is not mentioned in this paper): \(Q_k\) with importance \(\beta_4\).

In this stage of algorithm, again the OWA operator will be used to fuse the above mentioned scores. Figure 3 shows the mechanism which is used to this
will be computed with the following decision making function:

\[ D_k = \sum_{i=1}^{t} w_i B_i \]  

(9)

In which the vector \( B \) is a permutation of the vector \([\beta V_k, \beta U_k, \beta P_k, \beta Q_k] \) such that:

\[ \forall i > j; B_i \geq B_j ; i, j \in \{1, 2, 3, 4\} \]

Using this decision function a final score is assigned to each document and then all documents will be sorted according to their score. Then the final ranked list will be shown to the user. Figure 3 shows this process.

Agent starts the process with a heuristic selection of \( \beta \). For example:

\[ \beta_1 = 0.24, \beta_2 = 0.32, \beta_3 = 0.2, \beta_4 = 0.24 \]

But it will correct these coefficients as its life is going on. The related process will be explained in next section. Here again we use two rules to calculating a proper range for parameter \( a \) and then calculating weight vector.

**First rule:** If at least \( m \) criteria \((1 \leq m \leq 4)\) from \( V, U, P, Q \) have a score greater than \( y \), then the final score of that document must be greater than \( x \).

**Second rule:** If more than \( m' \) criteria \((1 \leq m' \leq 4)\) from \( V, U, P, Q \) have a score less than \( y' \), then the final score of that document must be less than \( x' \).

For \( m = 2, y = 0.8, x = 0.5 \) for first rule we have

\[
\ln\left(\frac{0.5}{0.8}\right) > a \Rightarrow a < 0.6781
\]

and for \( m' = 3, y' = 0.4, x' = 0.75 \) for the second rule:

\[
\ln\left(\frac{0.7 - 0.4}{0.6}\right) > a \Rightarrow a > 0.5
\]

Then if we choose \( a \) from \([0.5, 0.6781]\), the two above rules will be satisfied. With \( a = 0.58 \), which converts OWA to an Or-like (Optimistic) operator and with calculating final weights we have:

\[ W_1 = 0.4475; W_2 = 0.2215; W_3 = 0.1774; W_4 = 0.1537 \]

### 4.4 Learning Importance Coefficients

As the agent works more and more it must learn to do its task better. One way that agent can improve its abilities is to learn and update Importance Coefficients. \((0 \leq \beta_i \leq 1; i = 1, \ldots, 4)\)

There are two main methods to perform this learning. One way is get a feedback from user about the quality of results and updating \( \beta \) according to it. To doing this, methods like Q-Learning are suitable.

Another method is based on perception of agent from changes in values of \( V, U, P, Q \) and updating \( \beta \) according to them. In this paper we suggest a novel method to updating \( \beta \). The main idea is that in a complete period of searching process, the final score of documents is compared with each of \( V, U, P, Q \) and the \( \beta \) related to each of these scores which is nearest to final score will be increased. To achieve this goal we define an error measure at the time \( t \), \( e_i(t) \) that each time will be recalculated from new data, as follows:
Figure 3: Fusing of scores which are produced from figures 1 and 2, producing final ranked list, and updating importance coefficients and also behavioral model of servers.

\[
\begin{align*}
\epsilon_1^{(t+1)} &= \epsilon_1^t + \frac{1}{t+1} \sum_{k=1}^{N} |D_k - V_k| \\
\epsilon_2^{(t+1)} &= \epsilon_2^t + \frac{1}{t+1} \sum_{k=1}^{N} |D_k - U_k| \\
\epsilon_3^{(t+1)} &= \epsilon_3^t + \frac{1}{t+1} \sum_{k=1}^{N} |D_k - P_k| \\
\epsilon_4^{(t+1)} &= \epsilon_4^t + \frac{1}{t+1} \sum_{k=1}^{N} |D_k - Q_k|
\end{align*}
\]  

In this formulation, the index \( t \) shows the number of using agent from the reset time. So the smaller value for this error shows the related criterion is more suitable to deciding about documents.

The updating mechanism for \( \beta_i \) is:

\[
\beta_i^{(t+1)} = \frac{1 - \epsilon_i^{(t+1)}}{\sum_{j=1}^{4} (1 - \epsilon_j^{(t+1)})}
\]

Thus the importance coefficients will be adjusting over passing time.

5 Conclusion

In this paper we tried to study a part of an intelligent agent which acts as a customized Meta Search engine for user. In the mentioned architecture the agent performs fusion in data, feature, and decision levels. In sections 4.1 and 4.2, the agent fuses data about number of n-grams and distribution of them to produce features and in section 4.3 it fuses 4 different features to make a decision. The designed structure for agent is able to use other well-known fusion methods rather than OWA. In next steps authors will try to implement other methods and compare the results.

References


