CHAPTER 3

DYNAMIC NOMINAL LANGUAGE MODEL FOR INFORMATION RETRIEVAL

3.1 INTRODUCTION

Information Retrieval (IR) models produce ranking functions which assign scores to documents regarding a given query and this consists of two tasks: The task representing documents and query and the task which computes each document rank. IR systems build index terms to index and retrieve documents. Usually most index terms are any keyword which appears in text document collections. Most users have no opinion/training in query formation to retrieve results. To retrieve answers to a query, IR system predicts documents users find relevant/irrelevant. Predicting function is a ranking algorithm to ensure ordering of retrieved documents.

Language modelling approach to retrieval performs well. An advantage of this new approach is its statistical foundation. This approach to text retrieval was introduced by Ponte and Croft (1998) and later explored by (Song et al 1999; Allan et al 2002; Balog et al 2009). Language modelling approach’s simplicity and effectiveness with the fact that it leverages statistical methods developed in speech recognition and other areas ensures its attraction in developing new text retrieval methodology.
3.2 LANGUAGE MODELLING IN IR

Simple language models incorporate document and collection statistics systematically than Term Frequency–Inverse Document Frequency (tf.idf) based techniques. Language models and classical models using tf.idf work well, but further improvements require more techniques in addition to language modelling. Language modelling approach is shared with classical probabilistic approaches to IR, in that probabilistic modelling is taken as the primary scientific tool. Currently, this is a promising framework to advance information retrieval to meet challenges from diverse data sources/advanced retrieval tasks.

Language Model defines documents probability distribution using them to predict likelihood of query terms observation. Language model has been defined for all documents and it is used to inquire about chances of query generation. Some commonly used models are as:

3.2.1 Language Model based on Markov process:

Language model uses Markov process to assign occurrence probability to a word sequence ‘S’ as follows

$$P_d(S) = \prod_{i=1}^{n} P(K_i|K_{i-1}, K_{i-2}, ..., K_{i-(n-1)})$$

(3.1)

\[ n \] - order of the Markov process

if \( n = 1 \), then the model is called Unigram model.

3.2.2 Language Model based on Bernoulli process:

If index terms independence is assumed, then \( P(q|M_j) \) has been computed using Multi-variable Bernoulli process.
\[ P_{\text{d}}(K_{i} | M_{j}) = \prod_{k \in q} P(K_{i} | M_{j})^* \prod_{k \in q} P(K_{i} | M_{j}) \]  
(3.2)

The simple estimate of term probabilities

\[ P(K_{i} | M_{j}) = \frac{f_{i,j}}{\sum_{i} f_{i,j}} \]  
(3.3)

### 3.2.3 Language Model based on Multi-nominal process

If terms’ being independent among them is assumed, probability scoring function is defined as

\[ P(q | M) = \prod_{K \in q} P(K | M) \]  
(3.4)

By taking the log on both sides

\[ \log P(q | M) = \sum_{K \in q} \log P(K | M) \]  
(3.5)

\[ = \sum_{K \in q \cap d_{j}} \log P(K | M) = \sum_{K \in q \cap d_{j}} \log P(K | M) \]  
(3.6)

### 3.2.4 Language model based on Poisson Distribution

On the Poisson setup,

\[ P(d | R) = \prod_{k \in q} \frac{e^{-\lambda_{k} d_{k}} \lambda_{k}^{d_{k}}}{d_{k}!} \]  
(3.7)

\[ P(d | \neg R) = \prod_{k \in q} \frac{e^{-\mu_{k} d_{k}} \mu_{k}^{d_{k}}}{d_{k}!} \]  
(3.8)
The similarity meantime is derived as

\[
Sim(d, q) = O(R \mid d) = \frac{P(R \mid d)}{P(\neg R \mid d)}
\]

\[
Sim(d, q) = \sum_{k \in q} d_i \log \lambda_i - d_i \log \mu_i
\]

(3.9)

The maximum likelihood estimation for the Poisson distribution

\[
\mu_i = \frac{1}{n} \sum_{i=1}^{n} d_i
\]

(3.10)

For n documents with values \(d_i\) as word frequencies

### 3.2.5 Smoothing

Using estimate smoothing is necessary to avoid over-fitting (believing information given by small observed sample). Smoothing estimates and accounts for unseen terms in relevant/non-relevant documents.

### 3.2.6 Query Likelihood Model

Query likelihood model is based on query generation probability in document language model

\[
Score(Q, D) = \prod_{i=1}^{n} P(q_i \mid \theta_D)
\]

(3.11)

\(q_i\) denotes \(i^{th}\) query term.

Using linear interpolation smoothing query likelihood model sources documents as

\[
Score(Q; D) : \prod_{i=1}^{n} \lambda P(q_i \mid \theta_D) + (1 - \lambda) P(q_i \mid \theta_s)
\]

(3.12)
where $\theta_c$ denotes the collection language model and $\lambda$ denotes the mining coefficient.

### 3.3 QUERY LIKELIHOOD RETRIEVAL METHOD

The query likelihood retrieval method (Ponte et al 1998) enjoyed success in many retrieval tasks (Zhai et al 2001). It assumes that a query is a sample from a language model: given a query $Q$ and a document $D$, the likelihood of “generating” query $Q$ with a model estimated based on document $D$ is assumed. Documents based on likelihood of generating query are assumed.

Though query likelihood performed well, its theoretical foundation (Robertson et al 2001; Jones et al 2001) was criticised. Lafferty and Zhai (2002) proved that under certain assumptions, query likelihood retrieval method was justified based on probability ranking principle (Robertson et al 1977) and being regarded as probabilistic retrieval models foundation.

In query likelihood retrieval (Ponte et al 1998), given a query $Q$ and a document $D$, the likelihood of “generating” query $Q$ with a model $\theta_D$ estimated based on document $D$, is computed and then the document, based on its query likelihood is ranked:

\[
Score(D, Q) = p(Q | \theta_D)
\]  
(3.13)

The query generation is based on any language model (Miller et al 1999; Zhai et al 2001; Metzler et al 2004; Mei et al 2007; Tsagkias et al 2011). Till date, using a multinomial distribution (Miller et al 1999; Zhai et al 2001) for $\theta_D$ was popular and successful. With multinomial distribution, query likelihood is
where \( c(w, Q) \) is count of term \( w \) in query \( Q \). According to maximum likelihood estimator, the following estimation of document language model \( \theta_D \) for multinomial model is ready:

\[
p(w|\theta_D) = \prod_w p(w|\theta_D)^{c(w,Q)}
\]  

(3.14)

where \( c(w, D) \) indicates frequency of \( w \) in document \( D \), and \(|D|\) is document length. \( \theta_D \) needs smoothing to offset zero-probability problem, with an effective method being the Dirichlet prior smoothing (Zhai, et al., 2001):

\[
p(w|\theta_D) = \frac{|D|}{|D| + \mu} p_{ml}(w|D) + \frac{\mu}{|D| + \mu} p(w|C)
\]  

(3.16)

Here \( \mu \) is the smoothing parameter (Dirichlet prior), and \( p(w|C) \) is the collection language model which is estimated as

\[
p(w|C) = \frac{c(w,C)}{\sum_w c(w,C)}
\]  

(3.7)

where \( c(w, C) \) indicates the count of term \( w \) in the whole collection \( C \).

Query likelihood scoring function ranks documents using following formula (Zhai et al 2001):

\[
\log p(Q|\theta_D) = \sum_{w \in Q} c(w, Q) \log \left( 1 + \frac{c(w, D)}{\mu p(w|C)} \right) + |Q| \log \frac{\mu}{|D| + \mu}
\]  

(3.18)

where \( |Q| \) represents query length.
3.3.1 Query Reformulation and Expansion

To fine-tune/revise the query, two techniques consider relevant feedback:

(i) Query expansion adding new query terms to query from high relevant document

(ii) Term Reweighting: This increases terms weight in relevant documents and decreases terms weight those irrelevant.

3.3.2 Query Reformulation Algorithm

Revise Query vectors using vector Algebra:

(i) Add vectors for relevant documents to query vector.

(ii) Subtract vectors for irrelevant document from query vector.

This process adds positive/negative weighted terms to query & reweights initial query terms.

3.4 NOMINAL LANGUAGE MODEL (NLM)

Nominal Language Model (NLM) based language modelling goes with part of speech of a given query’s literal language, constituting factors with noun and adjectives. Informational query attempts to capture a document with data, relevant to analysis area. NLM based Information Retrieval process is an efficient method to extract relevant documents. Language modelling is processed with natural language processing methods.
3.5 RETRIEVAL METRICS - PRECISION AND RECALL

Precision and Recall metrics evaluate a set of un-ranked results and consider differences between documents set retrieved for a query and documents set relevant to a user’s need. Trade-off between Precision and Recall can also be user specific. Some users care about precision/recall, without asking users, how a search engine can guess whether a specific user cares more about precision than recall or vice versa.

\[
\text{recall} = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of relevant documents}}
\]

\[
\text{precision} = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of documents retrieved}}
\]

**Figure 3.1 Retrieval metrics**

If Information request \( I \) and its set \( R \) of relevant documents, answers \( I \) set \( A \), then precision and recall measures are as follows. Precision is retrieved documents fraction (A) relevant to

\[
\text{Precision} = P = \frac{|R \cap A|}{|A|}
\]

(3.19)

Recall is relevant documents(R) fraction determined

\[
\text{Recall} = r = \frac{|R \cap A|}{|R|}
\]

(3.20)
3.6 FLOW CHART OF THE PROPOSED METHODOLOGY

Flow chart of the proposed technique is shown in Figure 3.2.

Figure 3.2 Flowchart of the Proposed Methodology
The following sections details the various techniques used in the proposed method.

3.6.1 Datasets

The different types of Dataset include:

3.6.1.1 Reuters-21578 Dataset

The Reuters-21578 Text Categorization Test Collection is a standard text categorization benchmark having 21578 Reuters news documents from 1987, all labelled manually. Labels are of five different categories like 'people', 'places' and 'topics'. Total categories are 672 with many occurring very rarely. The Reuters-21578 data set is commonly used for newswire stories categorization into hand-labelled topics. Each news story is hand-labelled with some topic labels like “corn”, “wheat” and “corporate acquisitions”. Some topics overlap and hence belong to more than one category. The 12902 articles from “ModApte” split of the data five were used and, to stay comparable with previous studies, top ten most frequent topics were considered. Reuter’s collection is distributed in 22 files beginning with a document type declaration:

```xml
<DOCTYPElewis SYSTEM “lewis.dtd”>

Each article starts with an “open tag” of the form

```xml
<REUTERS TOPICS=?? LEWISSPLIT=?? CGISPLIT=?? OLDID=?? NEWID=??>
```

where the ?? are filled appropriately. Each article ends with the form’s “close tag.”:

```xml
</REUTERS>
```
Each REUTERS tag specifies values of five attributes: TOP-ICS, LEWISSPLIT, CGISPLIT, OLDID, and NEWID which identify documents and document groups. Attribute values determine how documents are divided into training and test sets. The experiments in this work used modified Apte split, most used in literature.

Each document represents a stemmed, TF-IDF weighted word frequency vector with each vector having unit modulus. A common words stop list was used with words occurring in less than three documents being ignored.

3.6.1.2 MovieLens dataset

Group Lens Research (http://www.grouplens.org/node/73) gives many collections of movie ratings data gathered from users of MovieLens in late 1990s and early 2000s. Data provides movie ratings, movie metadata (genres/year), and users demographic data (age, zip code, gender, occupation). The second dataset is provided by Movie-Lens research project, a web–based movie recommender system that began in 1997 and includes 943 users, 1,682 movies and has 100,000 transactions totally. Ratings are based on a 1 to 5 scale. The archive has many files, with a list of movie IDs and titles, and u. data, containing actual ratings in the following format:

196 242 3  881250949
186 302 3  891717742
22 377 1  878887116
244 51 2  880606923
166 346 1  886397596
298 474 4  884182806
3.6.2 **Inverse Document Frequency (IDF)**

Inverse Document Frequency (IDF) is a numerical statistic reflecting the importance of a word to a document in a collection/corpus, often used as a weighting factor in information retrieval/text mining. IDF value increases in proportion to the many times a word appears in the document, and offset by the word’s frequency in the corpus. This helps control the fact that some words are more common than others. IDF weighting scheme variations are used by search engines as a central tool in scoring/ranking a document's relevance regarding a user query. IDF is successfully used to filter stop-words in various fields like text summarization/classification. The latter is a semi-supervised, machine learning task which automatically assigns a document to a set of pre-defined categories based on textual content and extracted features.

Inverse Document Frequency (IDF) is a measure of word’s importance and usually appears in many heuristic measures in information retrieval. Till now IDF has been a heuristic, a popular measure of a word’s importance and defined as the logarithm of the ratio of documents number having a given word. This ensures that rare words have high IDF and common words like “the” have low IDF which measures a word’s ability to discriminate between documents. Text Classification assigns a text document to a pre-defined class set automatically, through machine learning. Classification is on the basis of significant words/key-features of a text document. As classes are pre-defined it becomes a supervised machine learning task.

The term document frequency is computed as follows for a set of documents \( x \) and a set of terms \( a \). Each document is modelled as a vector \( v \) in the \( a \) dimensional space \( \mathbb{R}^a \). The term frequency denoted by \( freq(x,a) \),
expresses the number of occurrence of the term \( a \) in document \( x \). The term-frequency matrix \( TF(x,a) \) measures term association \( a \) with regard to a given document \( x \). \( TF(x,a) \) is assigned zero when the document has no term and \( TF(x,a)=1 \) when term \( a \) occurs in the document \( x \) or uses a relative term frequency which is term frequency as against the total occurrences of all document terms. Frequency is generally normalized by (Liu, et al., 2007):

\[
    TF(x,a) = \begin{cases} 
        0 & \text{freq}(x,a)=0 \\
        \frac{\text{freq}(x,a)}{1 + \log(1 + \log(\text{freq}(x,a)))} & \text{otherwise} 
    \end{cases} \quad (3.21)
\]

<table>
<thead>
<tr>
<th>Document/term</th>
<th>( t_1 )</th>
<th>( t_2 )</th>
<th>( t_3 )</th>
<th>( t_4 )</th>
<th>( t_5 )</th>
<th>( t_6 )</th>
<th>( t_7 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( d_1 )</td>
<td>0</td>
<td>5</td>
<td>12</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( d_2 )</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>19</td>
<td>0</td>
<td>1</td>
<td>24</td>
</tr>
<tr>
<td>( d_3 )</td>
<td>12</td>
<td>15</td>
<td>4</td>
<td>0</td>
<td>15</td>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>( d_4 )</td>
<td>23</td>
<td>4</td>
<td>9</td>
<td>3</td>
<td>12</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>( d_5 )</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>2</td>
<td>9</td>
<td>9</td>
<td>7</td>
</tr>
</tbody>
</table>

**Figure 3.3** Term frequency matrix showing frequency of terms per document

IDF represents scaling factor. When a term \( a \) occurs frequently in many documents, its importance is then scaled down because of its lowered discriminative power. The \( IDF(a) \) is defined as follows:

\[
    IDF(a) = \log \frac{1 + \sqrt{|X|}}{x_a} \quad (3.22)
\]

\( x_a \) is the set of documents containing term \( a \).
TF-IDF usually uses text categorisation metric having two scores, term frequency and inverse document frequency. Term frequency is counting the times a term occurs in a document, while inverse document-frequency is attained by dividing total documents by documents where a specific word appears repeatedly. The multiplication of values results in a high score for frequently occurring words in limited documents. A low score is meant for terms appearing frequently in all documents.

Similar documents will have same relative term frequencies measured among a document set or between document and query. Cosine measure helps to locate similarity between documents. The cosine measure is given by:

\[
sim(v_1, v_2) = \frac{v_1 \cdot v_2}{\|v_1\| \|v_2\|}
\]

(3.23)

where \( v_1 \) and \( v_2 \) are two document vectors, \( v_1 \cdot v_2 \) defined as \( \sum_{i=1}^a v_{1i}v_{2i} \), and \( \|v_i\| = \sqrt{v_i \cdot v_i} \).

(3.24)

Figure 3.4 illustrates the relation between \( df_i \) and \( idf_i \) for a total of million documents.

<table>
<thead>
<tr>
<th>Term</th>
<th>( df_i )</th>
<th>( idf_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>brilliant</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>good</td>
<td>1000</td>
<td>3</td>
</tr>
<tr>
<td>under</td>
<td>10000</td>
<td>2</td>
</tr>
<tr>
<td>with</td>
<td>100000</td>
<td>1</td>
</tr>
<tr>
<td>the, movie</td>
<td>1000000</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 3.4 Relation between the \( df_i \) and \( idf_i \) for a total of million documents
3.6.3 Proposed tf/idf with Concept Expansion

A term/phrase can have many meanings, while a domain specific concept is unambiguous. It is useful to use the domain specific concepts in documents than terms to retrieve documents from a specific domain. Hence, the list of concepts in documents is extracted and a list of concepts annotates them. Too ensure this, the meaning of a term needs to be disambiguated to identify the concept it refers to. More than one term may refer to the same concept in some cases. Then concept frequency will include frequencies of the concept’s synonymous terms in the document (Roy et al 2008).

This rarely occurs in isolation. When a concept is important for a document, it usually has other related concepts. For example, ‘charge’ has at least two distinct meanings: electric charge and financial charge. When a document speaks about electric charge, the document will have other terms like current, electricity, etc. while in the case of financial charge; document will have terms like payment, amount, etc. The idea is to score a concept by looking at it and references to related concepts.

A list of terms and frequencies exists for each document. How each term in document is mapped to its corresponding concept and how each concept’s significance is estimated regarding the current document are discussed. An associated set of concepts for each term is obtained from ontology. A term maps to one or more concepts. As explained earlier the term ‘charge’ can map to electric charge, financial charge or criminal charge. Out of mapped concepts, the most appropriate concept for a specific domain should be located. Related concepts occurrences are looked into for this. Captured inter-concept relationship in ontology is used.
Figure 3.5 reveals a portion of concept graph in the ontology’s physics domain (Roy et al. 2010). A concept becomes significant when the document has many related concepts of that particular term. The proposed algorithm uses document terms with their frequency as input returning a concepts list with their significance regarding the document.

![Concept Graph](image)

**Figure 3.5 Relation among concepts**

The algorithm works as follows. For each term $t_i$ in the term list of a document $D$, the associated concepts $c_{ij}$ are obtained from the ontology. Let the impact of each associated concept $c_{ij}$ be $c_{ij}.impact$. The impact $c_{ij}.impact$ is initially taken as the normalized frequency of the term $t_i$ i.e. $t_i . frequency$. For each associated concept $c_{ij}$, the presence of the related concepts $rc_p$ in the document is taken into account. The impact of the associated concept $c_{ij}$ is then incremented by $\alpha*normalized \ term \ frequency$ for the occurrences of the terms $t_p$ corresponding to the concept $rc_p$. 
where $\alpha$ is the weight given to the related concepts. In this study, the value given is $\alpha = \frac{1}{2}$

For a particular term, a concept with maximum significance value is selected.

The algorithm is outlined below:

**Algorithm 1: Identification of Concept and its significance**

**Input:** $t_1, t_2, ..., t_n$ is the list of domain terms in the document $D$;

$\text{freq}_i$ is the normalized frequency of domain term $t_i$;

$num$ is the total number of tokens in the document $D$

**Output:** list of concepts $c_1, c_2, ..., c_m$ and their impact $c_i.\text{impact}$

```plaintext
for i ← 1 to n
{
    $\text{freq}_i$ ← $\frac{\text{freq}_i}{num}$
}

for i ← 1 to n
{
    $\text{concepts}_i$ ← \{c_{i_1}, c_{i_2}, ..., c_{i_k}\}
    $c_{i}.\text{impact}$ ← $\text{freq}_i$
}

for i ← 1 to n
{
    for j ← 1 to k
    {
        find the related concept $src_p$ of $c_{i_j} (rc_p$‘s corresponding term $tp)$ in $D$
    }
}
```
if \( t_p \) occurs in \( D \)

\[ c_y.impact \leftarrow c_y.impact + \alpha \times t_p.frequency / / \alpha = 1/2 \]

\}
}\)

for \( i \leftarrow 1 \) to \( n \)

\{
  if \( x.impact > \text{threshold} \)
  return \( x \)
  else
  return null
\}

The algorithm returns the list of concepts and their impact scores.

Word Net is a lingual database for English, the link language and is termed as an abounding lexical database for English constituting groups of nouns, verbs, adjectives and adverbs called synsets, contrived on conceptual semantic and lingual relations. A corpus with proposed concept expansion using Word Net is formed.

3.6.4 Dynamic Nominal Language Model (DNLM)

In this proposal, NLM is assembled with rate specifications and ratio calculations through use of probabilistic terms involving comparing query terms occurrence with data store using conditional probability theorem.

3.6.4 Dynamic Nominal Language Model (DNLM)

In this proposal, NLM is assembled with rate specifications and ratio calculations through use of probabilistic terms involving comparing query terms occurrence with data store using conditional probability theorem.
3.6.4.1 Stag Affinity Ratio

Query expansion follows calculation of stag affinity ratio. Conditional probability is applied for query term with each document term being subjected to many occurrences. This proposal conjugates semantic methods and Language Model Techniques. This conjugative IR method audits single word’s clear meanings in the query leading to results with accuracy. \( P(Q|T) \), called LM Coefficient \( (\gamma) \) determines each term’s occurrence number in each document. LM coefficient is evaluated by,

\[
P(Q|T_i) = \frac{a-b}{a}
\]  \hspace{1cm} (3.25)

where ‘i’ denotes terms number, ‘a’ represents each term’s co-occurrence number and ‘b’ stands on denoting the term. Similarly, each term’s weight corresponding to WordNet affinity is calculated and noted as ‘w’.

Stag affinity rate \( (\delta) \) is the product of individual word weight in the document and occurrences number of each term of a given query, termed as LM coefficient here.

\[
\delta_k = \sum_{j=1}^{m} \log \left( \prod_{i=1}^{n} [(\gamma_i, w_i)] \right)
\]  \hspace{1cm} (3.26)

where ‘i’ varies from \( 1 \leq i \leq n \), stands for query terms, ‘j’ ranges \( 1 \leq j \leq m \), denotes terms number in each document and ‘k’ represents documents number.

The affinity rate for each word is calculated and finally iterated for all words in the document. A document is related to query regarding
similarity, occurrence and relationship. To approximate stag affinity rate, its log value is first determined, thus making an effective evaluation criteria reducing fluctuations over high values determined by stag affinity rate. So, there is a possibility of participation and non-participation terms in user query regarding WordNet, a lexical English database. To improve the proposed algorithm’s consistency based on IR, association participation and non-participation rate terms of WordNet regarding user query is evaluated. Participation terms go with remaining probabilistic measurements there being a necessity to determine non-participating terms ratio of each document’s query. Non-Participating terms count evaluation reduces effects on accurate results. Association between non-participation and participation terms are evaluated through application of Kendal Coefficient. Stag affinity rate for non-participation terms will be 0. When the Kendal coefficient equation results are negative, it reveals higher non-participation terms in user query. Association between participation and non-participation word is termed elusion phrase ratio ($\tau$). Kendal coefficient is a non-parametric statistic with concordance.

Generally, Kendal coefficient ranges from 0 to 1 and value assumed regarding probabilistic distribution values. Kendal’s coefficient elusion phrase ratio is given by,

$$ Elusion Phrase Ratio (\tau_k) = \left( \frac{P \cdot NP}{P + NP} \right) / \alpha $$

(3.27)

where P represents participation terms and NP represents non-participation term regarding WordNet and $\alpha$ here is an arbitrary constant implying on range between $0 \leq \alpha \leq 1$. This calculation is biased for query terms accuracy determination.
3.6.4.2 Score Specific Ratio

Dependency between LM Coefficient and term weights is applied to correlation statistics to compute score specific ratio ($\rho$) value. Computation is through application of Pearson correlation coefficient to participating items numbers. Determination of $\rho$ is given as,

$$\frac{\sum_{i=1}^{n} (w_i - \bar{w})(v_i - \bar{v})}{\sqrt{\sum_{i=1}^{n} (w_i - \bar{w})^2 \sum_{i=1}^{n} (v_i - \bar{v})^2}}$$

(3.28)

where, correlation coefficient between weight and LM coefficient $\rho_{(w,v)}$ is conferred as,

Score Specific Ratio = $\rho_{(w,v)}$

The correlation calculation defines dependence statistical relationship between measured weight and LM coefficient value. Correlation coefficient imposes dependency degree between given terms against document. The score specific ratio produces a tedious value. Mean correlation value is obtained from $\rho$ of terms in a given query individually being applied for net-affinity rate evaluation. It ensures unvarying correlation value for more estimates. Thus, Monotonousness is evaluated by this correlation calculation. Determination of score specific ratio is an important part of affinity based Nominal Language Model, specifying correlation between weight of words in the document set and the Language Model Coefficient for excerpting number of words occurrences of user query against document set.

3.6.4.3 Net-Affinity Rate

Net-affinity rate is calculated through use of stag affinity rate, calculated to determine query terms similarity rate regarding WordNet, score
specific ratio specifying correlation between word weight and LM coefficient and elusion phrase ratio for determining participation and non-participation terms association.

Net-affinity rate is described further by ratio of stag affinity rate and score specific ratio with destruction of elusion phrase ratio, mentioned (3.29).

\[ \sigma_{k=1} = \left[ \frac{\hat{p}_k}{\rho_k} \right] \cdot \tau_k \]  

where ‘k’ represents the document sets and varies as \( 0 \leq k \leq r \).

Net-Affinity Rate finds similarity ratio for terms in user query with classifier having all relevant documents. The results sum up similarity ratio for all terms in query with entire document set. Terminal calculation of the proposed method is Net-affinity rate evaluation, summarizing all similarities and correlates results. The outcome is documents with greater accuracy relevant to user query which are ranked, based on scores for efficient result display.

The concept extraction module identifies each document’s concept; this is being done through an ontology collection. Terms are matched to the ontology’s concepts, synonyms, metonyms and hyponyms. Concept weight aka semantic weights are estimated through the concept and its element count. A semantic cube is constructed using concepts and semantic weight. The latter is calculated for a term, the weight being calculated on the number of occurrences. It is then multiplied with 100% weight for a concept. Semantic weights of other relations are calculated by equation.
Association rule is finally applied to words group to find a frequent item set. Agrawal, et al., (1993) provided an association rule mining statement for transaction databases. Let I = \{i_1, i_2, \ldots, i_m\} be the universe of items. A set X of items is an itemset. A transaction t = (tid, X) is a tuple where tid is a unique transaction ID and X an itemset. A transaction database D is a transactions set. The count of an item set X in D, denoted by count(X), is the number of transactions in D containing X. The support of an itemset X in D, denoted by \(25 \text{ supp}(X)\), is transactions proportion in D containing X. The rule X -> Y holds in the transaction set D with confidence c where \(\text{conf}(X \rightarrow Y) = \frac{\text{supp}(XY)}{\text{supp}(X)}\), where supp(XY) denotes support of items X and Y occurring together. Association rule mining aims to retrieve all rules of the form X -> Y where \(\text{supp}(XY) > s\) and \(\text{conf}(X \rightarrow Y) > c\), with s and c being Learner-supplied thresholds on minimum support and minimum confidence respectively (Agrawal, et al., 1993). Association rules are built by locating frequent itemsetshaving a support greater than defined threshold. Based on this, rules with a minimum confidence are selected.

3.7 EXPERIMENTAL SETUP AND RESULTS

Reuters dataset evaluates the proposed methods. Query expansion involves evaluation of user query with regard to information retrieval methodologies. Query expansion process is followed by calculations with nominal language modelling. NLM involves determination of Net-affinity rate comparing user query with WordNet data store through calculations with...
conditional probability of conceits. WordNet includes English language related data which provides user queries. It constitutes synsets which define short, common descriptions and store various semantic correlations between synonym sets. Elusion phrase ratio is then calculated to find the query’s non participation terms against WordNet data store. The weight of each user query is determined on its occurrences/importance, with score ratio being calculated with evaluated weight and the language model coefficient. Association rules are found.

Precision values for various techniques for MovieLens dataset and Reuters dataset are evaluated. The techniques used were tdf.idf, Language modelling using query likelihood, proposed concept expansion and proposed DNLM. The experimental results for MovieLens dataset for precision and F measure are tabulated in Table 3.1 and Table 3.2 respectively. Figure 3.6 and 3.7 show the same.

**Table 3.1 Precision values for various techniques for MovieLens dataset**

<table>
<thead>
<tr>
<th>Recall</th>
<th>TDF-IDF</th>
<th>Concept expansion</th>
<th>Language modelling using Query likelihood</th>
<th>DNLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0.6912</td>
<td>0.7321</td>
<td>0.7128</td>
<td>0.793</td>
</tr>
<tr>
<td>0.1</td>
<td>0.633</td>
<td>0.6912</td>
<td>0.6732</td>
<td>0.7487</td>
</tr>
<tr>
<td>0.2</td>
<td>0.614</td>
<td>0.625</td>
<td>0.632</td>
<td>0.677</td>
</tr>
<tr>
<td>0.3</td>
<td>0.612</td>
<td>0.618</td>
<td>0.606</td>
<td>0.6694</td>
</tr>
<tr>
<td>0.4</td>
<td>0.5623</td>
<td>0.593</td>
<td>0.572</td>
<td>0.6423</td>
</tr>
<tr>
<td>0.5</td>
<td>0.541</td>
<td>0.572</td>
<td>0.558</td>
<td>0.6196</td>
</tr>
<tr>
<td>0.6</td>
<td>0.492</td>
<td>0.568</td>
<td>0.542</td>
<td>0.6153</td>
</tr>
<tr>
<td>0.7</td>
<td>0.4602</td>
<td>0.534</td>
<td>0.521</td>
<td>0.5784</td>
</tr>
<tr>
<td>0.8</td>
<td>0.421</td>
<td>0.501</td>
<td>0.49</td>
<td>0.5427</td>
</tr>
<tr>
<td>0.9</td>
<td>0.4101</td>
<td>0.464</td>
<td>0.432</td>
<td>0.5026</td>
</tr>
<tr>
<td>1</td>
<td>0.2308</td>
<td>0.3911</td>
<td>0.362</td>
<td>0.321</td>
</tr>
</tbody>
</table>
From the Figure 3.6 it is seen that precision values for DNLM is higher than the precision values for all the other techniques. The precision values for DNLM is higher than the precision values for tdf.idf by 14.7% when recall is 0.01 and 22.5% when recall is 0.9. DNLM performs better than concept expansion by 8.3% when recall is 0.01 and 0.9. Similarly for Language modelling using query likelihood, DNLM performs better by 11% when recall is 0.01 and 16% when recall is 0.09.
Table 3.2  Average F measure values for various techniques for MovieLens dataset

<table>
<thead>
<tr>
<th></th>
<th>TDF-IDF</th>
<th>Concept expansion</th>
<th>Language modelling using Query likelihood</th>
<th>DNLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>F measure</td>
<td>0.507972</td>
<td>0.533996243</td>
<td>0.526405368</td>
<td>0.550098</td>
</tr>
</tbody>
</table>

Figure 3.7  Average F measure values for various techniques for MovieLens dataset

From the Figure 3.7 it is seen that average f measure values for DNLM is higher than the average f measure values for all the other techniques. The average f measure values for DNLM is higher than the average f measure values for tdf.idf by 8%, concept expansion by 3% and Language modelling using query likelihood by 4.5%.
The experimental results for Reuters-21758 dataset for precision and F measure are tabulated in Table 3.3 and Table 3.4 respectively. Figure 3.8 and 3.9 show the same.

Table 3.3  Precision values for various techniques for Reuters-21758 dataset

<table>
<thead>
<tr>
<th>Recall</th>
<th>TDF-IDF</th>
<th>Concept expansion</th>
<th>Language modelling using Query likelihood</th>
<th>DNLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0.7646</td>
<td>0.8099</td>
<td>0.7885</td>
<td>0.8772</td>
</tr>
<tr>
<td>0.1</td>
<td>0.7002</td>
<td>0.7646</td>
<td>0.7447</td>
<td>0.8282</td>
</tr>
<tr>
<td>0.2</td>
<td>0.6792</td>
<td>0.6914</td>
<td>0.6991</td>
<td>0.7489</td>
</tr>
<tr>
<td>0.3</td>
<td>0.677</td>
<td>0.6836</td>
<td>0.6704</td>
<td>0.7405</td>
</tr>
<tr>
<td>0.4</td>
<td>0.622</td>
<td>0.656</td>
<td>0.6328</td>
<td>0.7105</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5985</td>
<td>0.6328</td>
<td>0.6173</td>
<td>0.6854</td>
</tr>
<tr>
<td>0.6</td>
<td>0.5443</td>
<td>0.6283</td>
<td>0.5996</td>
<td>0.6807</td>
</tr>
<tr>
<td>0.7</td>
<td>0.5091</td>
<td>0.5907</td>
<td>0.5763</td>
<td>0.6398</td>
</tr>
<tr>
<td>0.8</td>
<td>0.4657</td>
<td>0.5542</td>
<td>0.542</td>
<td>0.6003</td>
</tr>
<tr>
<td>0.9</td>
<td>0.4537</td>
<td>0.5133</td>
<td>0.4779</td>
<td>0.556</td>
</tr>
<tr>
<td>1</td>
<td>0.2553</td>
<td>0.4326</td>
<td>0.4004</td>
<td>0.3551</td>
</tr>
</tbody>
</table>

Figure 3.8  Precision values for various techniques for Reuters-21758 dataset
The Figure 3.8 shows the relationship between precision and recall and it is seen that precision values for DNLM is higher when compared with the precision values for all the other techniques. The precision values for DNLM is higher than the precision values for tdf.idf by 14.7% when recall is 0.01 and 22.5% when recall is 0.9. DNLM performs better than concept expansion by 8.3% when recall is 0.01 and 0.9. Similarly for Language modelling using query likelihood, DNLM performs better by 11% when recall is 0.01 and 16% when recall is 0.09. Increasing recall conceivably increases the quality of the results with more relevant documents.

Table 3.4 Average F measure values for various techniques for Reuters-21758 dataset

<table>
<thead>
<tr>
<th></th>
<th>TDF-IDF</th>
<th>Concept expansion</th>
<th>Language modelling using Query likelihood</th>
<th>DNLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>F measure</td>
<td>0.532699</td>
<td>0.558495</td>
<td>0.550983</td>
<td>0.574389</td>
</tr>
</tbody>
</table>

Figure 3.9 Average F measure values for various techniques for Reuters-21758 dataset
It is apparent from the above pictorial representation that the proposed methodology produces higher average f measure than the existing tdf.idf method, concept expansion and Language modelling using Query likelihood. The average f measure values for DNLM is higher than the average f measure values for tdf.idf by 7.8%, concept expansion by 2.8 %, and Language modelling using query likelihood by 4.2%.

3.8 CONCLUSION

This work proposes a method for Information Retrieval based on Nominal Language Model supporting e-learning environments. A term/phrase has multiple meanings, while a domain specific concept can be unambiguous. It is useful and better to use documents domain specific concepts than retrieving documents terms which belong to a specific domain. Hence, list of concepts present in documents are extracted and annotated with the concepts lists. NLM based approach includes lexical resources of natural language processing where the process moves through data extraction with given query. Methods of conditional probability theorem were used to determine Affinity rates to ensure that this approach was persuasive. Experiments showed that proposed DNLM method achieved better precision compared to traditional tdf.idf and Language modelling using Query likelihood.