Available online at www.sciencedirect.com

ScienceDirect

Computer Speech and Language 23 (2009) 126–144

COMPUTER SPEECH AND LANGUAGE

www.elsevier.com/locate/csl

GA, MR, FFNN, PNN and GMM based models for automatic text summarization

Mohamed Abdel Fattah [a,b,*], Fuji Ren [a,c]

[a] Faculty of Engineering, University of Tokushima, 2-1 Minamijosanjima, Tokushima 770-8506, Japan
[b] FIE, Helwan University, Cairo, Egypt
[c] School of Information Engineering, Beijing University of Posts and Telecommunications, Beijing 100088, China

Received 13 September 2007; received in revised form 22 April 2008; accepted 28 April 2008
Available online 4 May 2008

Abstract

This work proposes an approach to address the problem of improving content selection in automatic text summarization by using some statistical tools. This approach is a trainable summarizer, which takes into account several features, including sentence position, positive keyword, negative keyword, sentence centrality, sentence resemblance to the title, sentence inclusion of name entity, sentence inclusion of numerical data, sentence relative length, Bushy path of the sentence and aggregated similarity for each sentence to generate summaries. First, we investigate the effect of each sentence feature on the summarization task. Then we use all features in combination to train genetic algorithm (GA) and mathematical regression (MR) models to obtain a suitable combination of feature weights. Moreover, we use all feature parameters to train feed forward neural network (FFNN), probabilistic neural network (PNN) and Gaussian mixture model (GMM) in order to construct a text summarizer for each model. Furthermore, we use trained models by one language to test summarization performance in the other language. The proposed approach performance is measured at several compression rates on a data corpus composed of 100 Arabic political articles and 100 English religious articles. The results of the proposed approach are promising, especially the GMM approach.
© 2008 Elsevier Ltd. All rights reserved.

Keywords: Automatic summarization; Genetic algorithm; Mathematical regression; Feed forward neural network; Probabilistic neural network; Gaussian mixture model

1. Introduction

With the huge amount of information available electronically, there is an increasing demand for automatic text summarization systems. Text summarization is the process of automatically creating a compressed version of a given text that provides useful information for the user (Ye et al., 2007; Steinberger et al., 2007; Dorr and

* Corresponding author. Address: Faculty of Engineering, University of Tokushima, 2-1 Minamijosanjima, Tokushima 770-8506, Japan. Tel.: +81 088 625 1545.
E-mail addresses: mohafi@is.tokushima-u.ac.jp (M.A. Fattah), ren@is.tokushima-u.ac.jp (F. Ren).

0885-2308/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.csl.2008.04.002

Gaasterland, 2007; Diaz and Gervás, 2007). Text summarization addresses both the problem of selecting the most important portions of text and the problem of generating coherent summaries. There are two types of summarization: extractive and abstractive. Extractive summarization methods simplify the problem of summarization into the problem of selecting a representative subset of the sentences in the original documents. Abstractive summarization may compose novel sentences, unseen in the original sources. However, abstractive approaches require deep natural language processing such as semantic representation, inference and natural language generation, which have yet to reach a mature stage nowadays (Ye et al., 2007).

Automated summarization dates back to the Fifties (Luhn, 1958). The different attempts in this field have shown that human-quality text summarization was very complex since it encompasses discourse understanding, abstraction, and language generation (Sparck, 1993). Simpler approaches were then explored that consist of extracting representative text-spans, using statistical techniques and/or techniques based on surface domain-independent linguistic analyses. Within this context, summarization can be defined as the selection of a subset of the document sentences which is representative of its content. This is typically done by ranking document sentences and selecting those with higher score and minimum overlap (Carbonell and Goldstein, 1998; Nomoto and Matsumoto, 2001). Most of the recent work in summarization uses this paradigm.

The process of text summarization can be decomposed into three phases: analysis, transformation, and synthesis. The analysis phase analyzes the input text and selects a few salient features. The transformation phase transforms the results of analysis into a summary representation. Finally, the synthesis phase takes the summary representation, and produces an appropriate summary corresponding to users' needs. In the overall process, compression rate, which is defined as the ratio between the length of the summary and that of the original, is an important factor that influences the quality of the summary. As the compression rate decreases, the summary will be more concise; however, more information is lost. While the compression rate increases, the summary will be larger; relatively, more insignificant information is contained. In fact, when the compression rate is 5–30%, the quality of the summary is acceptable (Hahn and Mani, 2000; Kupiec et al., 1995; Mani and Maybury, 1999; Yeh et al., 2005).

Because of the lack of powerful computers and difficulty in nature language processing (NLP), early work on text summarization focused on the study of text genres such as sentence position and cue phrase (Edmundson, 1969; Luhn, 1958). During the 1970s, artificial intelligence (AI) started to be applied (Azzam et al., 1999; McKeown and Radev, 1995; Schank and Abelson, 1977; Young and Hayes, 1985). AI exploited knowledge representations, such as frames or templates, to identify conceptual entities from a text and to extract relationships between entities by inference mechanisms. The major drawback is that limitedly defined frames or templates may lead to incomplete analysis of conceptual entities. During the 1990s, information retrieval (IR) was employed for the text summarization task (Aone et al., 1997; Goldstein et al., 1999; Gong and Liu, 2001; Hovy and Lin, 1997; Kupiec et al., 1995; Mani and Bloedorn, 1999; Salton et al., 1997; Teufel and Moens, 1997; Yeh et al., 2002). However, most IR techniques that have been exploited in text summarization focus on symbolic-level analysis, and they do not take into account semantics such as synonymy, polysemy, and term dependency (Hovy and Lin, 1997).

Recently many experiments have been conducted for the text summarization task. Some were about evaluation of summarization using relevance prediction (Hobson et al., 2007), ROUGEeval package (Sjöbergh, 2007), SUMMAC, NTCIR, and DUC (Over et al., 2007) and voted regression model (Hirao et al., 2007). Others were about single- and multiple-sentence compression using "parse and trim" approach and a statistical noisy-channel approach (Zajic et al., 2007) and conditional random fields (Nomoto, 2007). Other research includes multi-document summarization (Vanderwende et al., 2007; Harabagiu et al., 2007) and summarization for specific domains (Moens, 2007; Reeve et al., 2007; Ling et al., 2007).

In this work, sentences of each document are modeled as vectors of features extracted from the text. The summarization task can be seen as a two-class classification problem, where a sentence is labeled as "correct" if it belongs to the extractive reference summary, or as "incorrect" otherwise. We may give the "correct" class a value '1' and the "incorrect" class a value '0'. In testing mode, each sentence is given a value between '0' and '1' (values between 0 and 1 are continuous). Therefore, we can extract the appropriate number of sentences according to the compression rate. The trainable summarizer is expected to "learn" the patterns which lead to the summaries, by identifying relevant feature values which are most correlated with the classes "correct" or "incorrect". When a new document is given to the system, the "learned" patterns are used to classify each sentence of that document into either a "correct" or "incorrect" sentence by giving it a certain score value between '0' and '1'. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

The rest of the paper is organized as follows: Section 2 presents the different text feature parameters, Section 3 is about the proposed automatic summarization model, Section 4 shows the experimental results and finally Section 5 presents conclusions and future work.

2. Text features

1. f1 = Sentence position.We assume that the first sentences of a paragraph are the most important. Therefore, we rank a paragraph sentence according to its position in the paragraph and we consider maximum positions of 5. For instance, the first sentence in a paragraph has a score value of 5/5, the second sentence has a score 4/5, and so on. A score of zero is given to sentences paragraph position greater than 5 since position feature of these sentences is not significant.
2. f2 = Positive keyword in the sentence.Positive keyword is the keyword frequently included in the summary. It can be calculated as follows:

   $$Score_{f_2}(s) = \frac{1}{Length(s)} \sum_{i=1}^{n} tf_i * P(s \in S|keyword_i) \tag{1}$$

   where s is a sentence, n is the number of keywords in s, tf_i is the occurring frequency of $keyword_i$ in s. We divide the value by the sentence length to avoid the bias of its length (tf_i, n and Length(s) are calculated using sentence "s" from the testing data)

   $$P(s \in S|keyword_i) = \frac{P(keyword_i|s \in S)P(s \in S)}{P(keyword_i)}$$

   $$P(keyword_i|s \in S) = \frac{\#(sentence\,in\,summary, and\,contains\,keyword_i)}{\#(sentence\,in\,summary)}$$

   $$P(s \in S) = \frac{\#(sentence\,in\,training\,corpus,\ and\,also\,in\,summary)}{\#(sentence\,in\,training\,corpus)}$$

   $$P(keyword_i) = \frac{\#(sentence\,in\,training\,corpus\,,\ and\,contains\,keyword_i)}{\#(sentence\,in\,training\,corpus)}$$

   $P(s \in S|keyword_i)$ is calculated from the training corpus (manually summarized articles).
3. f3 = Negative keyword in the sentence.
   In contrast to f2, negative keywords are the keywords that are unlikely to occur in the summary. And it can be calculated as follows:

   $$Score_{f_3}(s) = \frac{1}{Length(s)} \sum_{i=1}^{n} tf_i * P(s \notin S|keyword_i) \tag{2}$$

4. f4 = Sentence centrality (similarity with other sentences).Sentence centrality is the vocabulary overlap between this sentence and other sentences in the document. It is calculated as follows:

   $$Score_{f_4}(s) = \left|\frac{Keywords\,in\,s \cap Keywords\,in\,other\,sentences}{Keywords\,in\,s \cup Keywords\,in\,other\,sentences}\right| \tag{3}$$

5. f5 = Sentence Resemblance to the title.Sentence resemblance to the title is the vocabulary overlap between this sentence and the document title. It is calculated as follows:

   $$Score_{f_5}(s) = \left|\frac{Keywords\,in\,s \cap Keywords\,in\,title}{Keywords\,in\,s \cup Keywords\,title}\right| \tag{4}$$

6. f6 = sentence inclusion of name entity (proper noun).Usually the sentence that contains more proper nouns is an important one and it is most probably included in the document summary. The score of f6 is calculated as follows:

$$Score_{f_6}(s) = \frac{\#(proper\,nouns\,in\,s)}{Length(s)} \tag{5}$$

7. f7 = sentence inclusion of numerical data.Usually the sentence that contains numerical data is an important one and it is most probably included in the document summary. The score of f7 is calculated as follows:

$$Score_{f_7}(s) = \frac{\#(numerical\,data\,in\,s)}{Length(s)} \tag{6}$$

8. f8 = sentence relative length.This feature is employed to penalize sentences that are too short, since these sentences are not expected to belong to the summary. We use the relative length of the sentence, which is calculated as follows:

$$Score_{f_8}(s) = length(s) * average\,sentence\,length \tag{7}$$

9. f9 = Bushy path of the node (sentence).The bushiness of a node (sentence) on a map is defined as the number of links connecting it to other nodes (sentences) on the map. Since a highly bushy node is linked to a number of other nodes, it has an overlapping vocabulary with several sentences and is likely to discuss topics covered in many other sentences (Salton et al., 1997). The Bushy path is calculated as follows:

$$Score_{f_9}(s) = \#(branches\,connected\,to\,the\,node) \tag{8}$$

The automatic method which is used to determine whether there is a link between two sentences is the similarity (vocabulary overlap) between these two sentences. The score of f9 is equal to this similarity divided by the longest sentence of them for normalization. We have used a threshold of 0.1 to discard branches among nodes that have similarities less than 0.1.

Fig. 1 shows a simplified example of such map (only 6 sentences of a certain document). As shown in the figure, the similarity (vocabulary overlap) between the first 2 sentences (S1 and S2) is 0.24. The highest bushy node is sentence S2 since it has 5 branches.Unlike LexRank feature, Bushy path is a simple and an effective text feature for single and multi-document summarization task. LexRank is used to compute sentence importance based on the concept of eigenvector centrality in a graph representation of sentences for multi-document summarization task (Erkan and Radev, 2004).

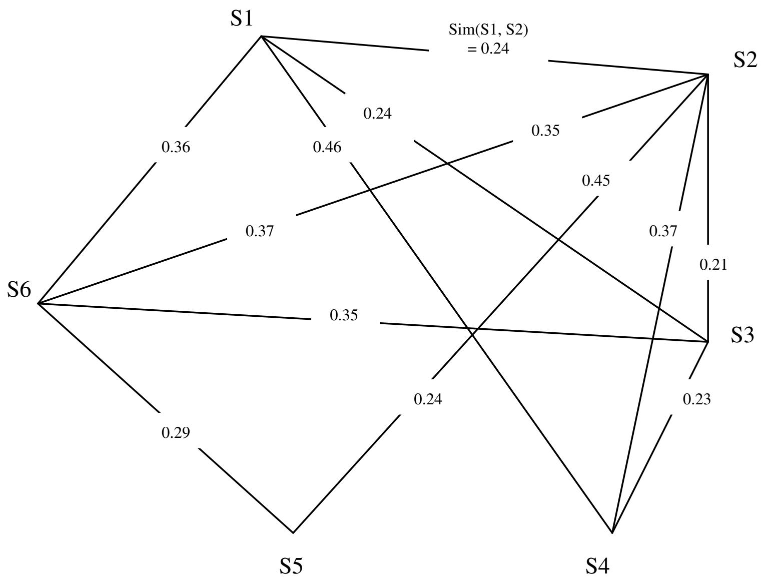

Fig. 1. A simplified example of Bushy map.

10. f10 = Summation of similarities for each node (aggregate similarity).This similarity is simply a vocabulary overlap between the 2 nodes (2 sentences) under consideration divided by the longest length of the 2 sentences (for normalization) as shown in Fig. 1.Aggregate similarity measures the importance of a sentence. Instead of counting the number of links connecting a node (sentence) to other nodes (Bushy path), aggregate similarity sums the weights (similarities) on the links. Aggregate similarity is calculated as follow:

$$Score_{f_{10}}(s) = \sum Node\,branch\,similarities \tag{9}$$

For instance, from Fig. 1, the $Score_{f_{10}} = 0.24 + 0.29 = 0.53$.

3. The proposed automatic summarization model

Fig. 2 shows the proposed automatic summarization model. We have two modes of operations:

1. Training mode where features are extracted from 100 manually summarized Arabic documents and 50 manually summarized English documents (50 not 100 to investigate the training data set size effect on summarization performance) and used to train GA, MR, FFNN, PNN and GMM models.
2. Testing mode, in which features are calculated for sentences from 100 Arabic and 100 English documents. (These documents are different from those that were used for training.) The sentences are ranked according to the sets of feature weights calculated during the training stage. Summaries consist of the highest-ranking sentences.

3.1. Genetic algorithm model (GA)

The basic purpose of genetic algorithms (GAs) is optimization. Since optimization problems arise frequently, this makes GAs quite useful for a great variety of tasks. As in all optimization problems, we are faced with the problem of maximizing/minimizing an objective function $f(x)$ over a given space X of arbitrary dimension (Russell and Norvig, 1995; Yeh et al., 2005). Therefore, GA can be used to specify the weight of each text feature.

For a sentence s, a weighted score function, as shown in the following equation is exploited to integrate all the ten feature scores mentioned in Section 2, where w_i indicates the weight of f_i.

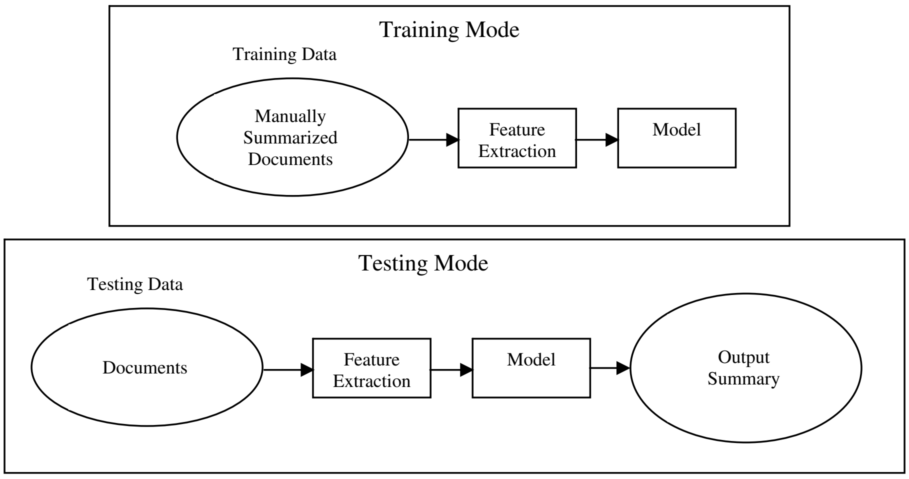

Fig. 2. The proposed automatic summarization model.

$$\begin{aligned} Score(s) = {} & w_1 \cdot Score_{f_1}(s) + w_2 \cdot Score_{f_2}(s) + w_3 \cdot Score_{f_3}(s) + w_4 \cdot Score_{f_4}(s) + w_5 \cdot Score_{f_5}(s) \\ & + w_6 \cdot Score_{f_6}(s) + w_7 \cdot Score_{f_7}(s) + w_8 \cdot Score_{f_8}(s) + w_9 \cdot Score_{f_9}(s) + w_{10} \cdot Score_{f_{10}}(s) \end{aligned} \tag{10}$$

The genetic algorithm (GA) is exploited to obtain an appropriate set of feature weights using the 100 manually summarized Arabic documents and the 50 manually summarized English documents. A chromosome is represented as the combination of all feature weights in the form of (w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9, w_{10}). Thousand genomes for each generation were produced. Evaluate fitness of each genome (we define fitness as the average precision obtained with the genome when the summarization process is applied on the training corpus), and retain the fittest 10 genomes to mate for new ones in the next generation. In this experiment, 100 generations are evaluated to obtain steady combinations of feature weights. A suitable combination of feature weights is found by applying GA.

For testing, a set of 100 Arabic documents and 100 English documents was used. We apply Eq. (10) after using the defined weights from GA execution. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

3.2. Mathematical regression model (MR)

Mathematical regression is a good model to estimate the text feature weights (Jann, 2005; Richard, 2006). In this model a mathematical function relates output to input. The feature parameters of the 100 manually summarized Arabic documents and the 50 manually summarized English documents are used as independent input variables and the corresponding dependent outputs are specified in the training phase. We try to get a relation between inputs and outputs to model the system. Then testing data are introduced to the system model for evaluation of its efficiency. In matrix notation, we can represent regression as follow:

$$\begin{bmatrix} Y0 \\ Y1 \\ \vdots \\ \vdots \\ Ym \end{bmatrix} = \begin{bmatrix} X01 & X02 & X03 & \cdots & X010 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ Xm1 & Xm2 & Xm3 & \cdots & Xm10 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ \vdots \\ w_{10} \end{bmatrix} \tag{11}$$

where $[Y]$ is the output vector, $[X]$ is the input matrix (feature parameters),$[W]$ is the linear statistical model of the system (the weights $w_1, w_2, \ldots w_{10}$ in Eq. (10)). m is the total number of sentences in the training corpus.

For testing, a set of 100 Arabic and 100 English documents was used. We apply Eq. (10) after using the defined weights from $[W]$. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

3.3. Feed forward neural network (FFNN)

Feed forward neural networks are good tools for classifications (Fattah et al., 2006b). Therefore, feed forward neural network can be used to classify a sentence as summary or not summary based on its features.

The layered structure of the neural network that we used is illustrated in Fig. 3. We have used 10 input units, 20 hidden units and 1 output unit to represent the neural network. Each input unit represents one input feature as described in Section 2.

All the 10 input features are represented by the feature vector $\overrightarrow{X}$. The input pattern $\overrightarrow{X}$ is propagated through the network in the following way.

The output of the hidden layer is given by

$$O_j^{(1)} = f\left(\sum_{k=1}^{N} W_{jk}^{(1)} X_k\right) \tag{12}$$

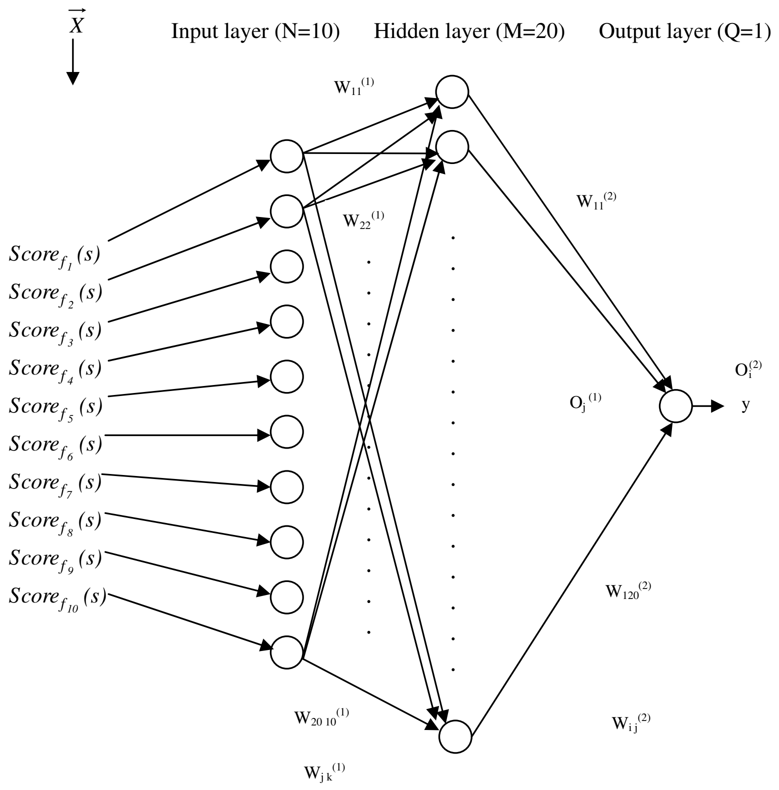

Fig. 3. The feed forward neural network structure.

where W_{jk} is the weight associated with the line between the input unit "k" and the hidden unit "j" f is a sigmoidal function given by

$$f(z) = \frac{1}{1+\exp(-z)} \tag{13}$$

The output of the output layer is given by

$$O_i^{(2)} = f\left(\sum_{j=1}^{M} W_{ij}^{(2)} O_j^{(1)}\right) \tag{14}$$

where W_{ij} is the weight associated with the line between the hidden unit "j" and the output unit "i".From Eqs. (12) and (14)

$$O_i^{(2)} = f\left(\sum_{j=1}^{M} W_{ij}^{(2)} f\left(\sum_{k=1}^{N} W_{jk}^{(1)} X_k\right)\right) \tag{15}$$

The mean squared error (MSE) is given by

$$E = \frac{1}{2}\frac{1}{G}\sum_{i=1}^{|G|}\sum_{\vec{x}\in G}[y_i(\vec{x}) - O_i^{(2)}(\vec{x})]^2 \tag{16}$$

where $y_i(\vec{x})$ is the desired neural network output value when the input is $\vec{x}$, G is the training data size (cardinality).

We differentiate Eq. (16) in order to minimize E (MSE). The change in weight for output unit i from hidden unit j is given by

$$\Delta W_{ij}^{(2)} = \frac{\partial E}{\partial W_{ij}^{(2)}} \tag{17}$$

$$= \sum_{\vec{x}\in G} [y_i - O_i^{(2)}] f'(O_i^{(2)}) O_j^{(1)} \tag{18}$$

where

$$f'(z) = \frac{\exp(-z)}{[1+\exp(-z)]^2} = f(z)(1-f(z)) \tag{19}$$

On the other hand,

$$\Delta W_{jk}^{(1)} = \sum_{\vec{x}\in G} \frac{\partial E}{\partial O_j^{(1)}} \frac{\partial O_j^{(1)}}{\partial W_{jk}^{(1)}} \tag{20}$$

$$\Delta W_{jk}^{(1)} = \sum_{\vec{x}\in G} \sum_i [y_i - O_i^{(2)}] f'(O_i^{(2)}) W_{ij}^{(2)} f'(O_j^{(1)}) \vec{x} \tag{21}$$

The new weight value is determined by the learning algorithm. The new weight is given by

$$W_{\text{new}} = W_{\text{old}} - \beta \Delta W \tag{22}$$

The output takes values between 0 and 1. The sentences that give high output values (tend to 1) are considered summary sentences. The sentences that give low output values (tend to 0) are not considered summary sentences. During test phase, all document sentences are ranked in a descending order according to their output scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

3.4. Probabilistic neural network (PNN)

Probabilistic neural networks are a versatile and efficient tool to classify high-dimensional data (Specht, 1990; Ganchev et al., 2003; Cain, 1990). The network weights and functions are backed by straightforward Bayesian probability, giving them an edge over other network models that have to be gradually optimized using techniques like gradient descent. Bayes' theorem can be used to perform probabilistically optimal classification as follows:

The probability distribution function (PDF) for a feature vector (X) to be of a certain category (class A for example as a summary sentence) is given by

$$f_{\text{a}}(X) = 1/(2\pi)^{p/2}\sigma^p(1/n_a)\sum_{i=1}^{n_{\text{a}}} \exp(-(X - Y_{\text{a}i})^\tau (X - Y_{\text{a}i})/2\sigma^2), \tag{23}$$

where $f_{\text{a}}(X)$ = the value of the PDF for class A at point X; X = test vector to be classified; i = training vector number; p = the training vector size; n_{a} = number of training vectors in class A; $Y_{\text{a}i}$ = ith training vector for class A; τ = transpose and σ = the standard deviation of the Gaussian curves used to construct the PDF.

Introducing a term to represent the relative number of trials in each category ($n_{\text{a}}/n_{\text{total}}$) simplifies the expression. Hence ($1/n_{\text{a}}$) term is canceled out as follows:

$$f_{\text{a}}(X) = 1/(2\pi)^{p/2}\sigma^p(1/n_{\text{total}})\sum_{i=1}^{n_{\text{a}}} \exp(-(X - Y_{\text{a}i})^\tau (X - Y_{\text{a}i})/2\sigma^2), \tag{24}$$

Terms common to all classes such as $1/(2\pi)^{p/2}$, σ^p and n_{total} can also be eliminated, leaving the following formula:

$$f_{\text{a}}(X) \alpha \sum_{i=1}^{n_{\text{a}}} \exp(-(X - Y_{\text{a}i})^\tau (X - Y_{\text{a}i})/2\sigma^2), \tag{25}$$

Hence the classifier can be expressed as follows:For a feature parameter X to belong to a category (r); the following formula must be verified:

$$\sum_i \exp(-(X - Y_{ri})^\tau (X - Y_{ri})/2\sigma^2) \geqslant \sum_i \exp(-(X - Y_{si})^\tau (X - Y_{si})/2\sigma^2) \tag{26}$$

where (s) represents the other category.

The expression $(X - Y_{ri})^\tau (X - Y_{ri}) = (X^\tau X) - (2X^\tau Y_{ri}) + (Y_{ri}^\tau Y_{ri}) = 1 - (2X^\tau Y_{ri}) + 1 = 2 - (2X^\tau Y_{ri})$ allowing formula (26) to be simplified as follows:

$$\sum_i \exp((X^\tau Y_{ri} - 1)/\sigma^2) \geqslant \sum_i \exp((X^\tau Y_{si} - 1)/\sigma^2), \tag{27}$$

Fig. 4 shows the structure of the P-NNT implementation. Each neuron in layer one receives an element of vector X ($x_1, x_2 \ldots x_n$) of a certain sentence to be classified as summary or not summary sentence, ($n = 10$ in our case). The weight matrix scaling these inputs is formed by the elements of the training vectors divided by the constant (σ^2). The first layer has a bias of $-1/\sigma^2$. The inputs of layer one are summed, producing ($X^\tau Y_{ri} - 1$). Then, this value is divided by σ^2 and the exponential transfer function is applied, resulting in outputs of $\exp((X^\tau Y_{ri} - 1)/\sigma^2)$ and $\exp((X^\tau Y_{si} - 1)/\sigma^2)$, where ($s$) represents the remaining category. The second layer has two neurons: each one is associated with a specific category of output mentioned before (summary category and not summary category). The inputs from the first layer of each category are summed to produce the expressions $\sum_{i=1}^{m} \exp((X^\tau Y_{ri} - 1)/\sigma^2)$ and $\sum_{i=1}^{m} \exp((X^\tau Y_{si} - 1)/\sigma^2)$. The output of each neuron represents the probability that the vector X belongs to its class. The neuron in layer 2 with the greatest output determines how the vector is classified.

3.5. Gaussian mixture model (GMM)

Sequential learning systems such as Hidden Markov Models have been exploited for the text summarization task, but they cannot fully exploit the rich linguistic features since they have to assume independence among the features for tractability (Conroy and O'Leary, 2001; Jing, 2002). A Markov Random Field (MRF) which is an undirected graphical model in which vertices represent variables and edges represent correlations between variables does not have this drawback. MRFs are used to specify the conditional probabilities of possible label sequences given an observation sequence. Moreover, the conditional probabilities of label sequences can depend on arbitrary, non independent features of the observation sequence (Kleinberg and Tardos, 2002; Zhu and Ghahramani, 2002).

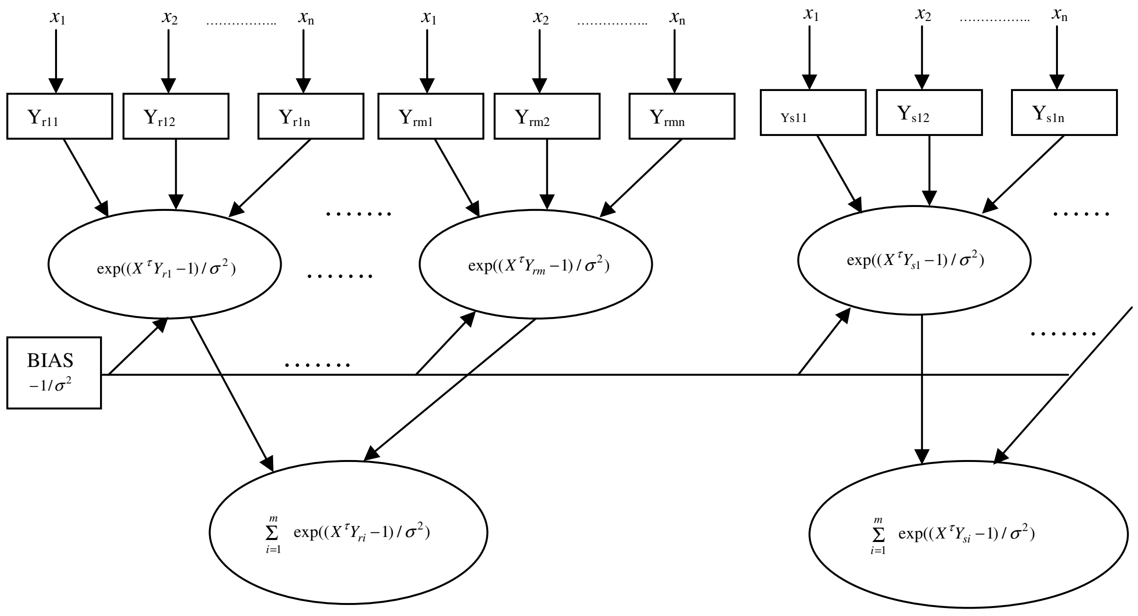

Fig. 4. The structure of PNN implementation.

The use of Gaussian Mixture models as a classification tool is motivated by the interpretation that the Gaussian components represent some general output dependent features and the capability of Gaussian mixtures to model arbitrary densities (Reynolds, 1995; Pellom and Hansen, 1998; Fattah et al., 2006a).

The probability density function for a certain class (category) feature vector X is a weighted sum, or *mixture*, of k class-conditional Gaussian distributions. For a given class model λc, the probability of observing X is given by

$$p(X|\lambda c) = \sum_{k=1}^{K} w_{c,k} N\left(X; \vec{\mu}_{c,k}, \sum\nolimits_{c,k}\right), \tag{28}$$

where $w_{c,k}$, $\vec{\mu}_{c,k}$, $\sum_{c,k}$ are the mixture weight, mean, and covariance matrix, respectively, for the ith component, which has a Gaussian distribution given by

$$N\left(X; \vec{\mu}, \sum\right) = \frac{1}{\sqrt{(2\pi)^n |\sum|}} \exp\left(-\frac{1}{2}(X - \vec{\mu})^{\tau} \sum\nolimits^{-1} (X - \vec{\mu})\right), \tag{29}$$

where n is the dimension of X. We used $\sum$ as diagonal covariance matrices. Given a set of training vectors of a certain class, an initial set of means is estimated using k-means clustering. The mixture weights, means, and covariances are then iteratively trained using the expectation maximization (EM) algorithm.

Using this approach, we constructed a class-dependent model for each category. After that we used all models for the summarization task using the maximum likelihood of each category as follows:

For a given set of class-dependent reference models $(\lambda 1, \lambda 2)$ and one feature vector sequence $X = \{x_1, x_2, \ldots, x_n\}$, the minimum error Bays' decision rule is

$$\text{arg max}_{1 \leqslant l \leqslant 2} p(\lambda l|X) = \text{arg max}_{1 \leqslant l \leqslant 2} \frac{p(X|\lambda l)}{p(X)} p(\lambda l), \tag{30}$$

Using formula (30), a feature vector sequence X may be classified as one of the two-classes (summary or not summary).

4. Experimental results

4.1. The Arabic and English data

Two hundred Arabic articles in the domain of politics and 150 English articles in the domain of religion were collected from the Internet archive. One hundred Arabic and 50 English articles were manually summarized using compression rate of 30%. These manually summarized articles were used to train the previously mentioned five models. The other 100 Arabic and 100 English articles were used for testing. The average number of sentences per Arabic and English articles is 26.8 and 32.7, respectively. Moreover, to investigate the proposed approaches performance on newswire data, we have exploited DUC 2001 for single document test.

We use an intrinsic evaluation to judge the quality of a summary based on the coverage between it and the manual summary. We measure the system performance in terms of precision from the following formula:

$$P = \frac{|S \cap T|}{|S|} \tag{31}$$

Recall would be calculated as $R = \frac{|S \cap T|}{|T|}$. However, we compare the candidate summaries to the reference summaries at the same compression ratio, so $|S| = |T|$, and precision = recall.

Where P is the precision, R is the recall, T is the manual summary and S is the machine-generated summary. Moreover, a 95% confidence interval was considered for all tests to estimate the statistical uncertainty.

4.2. Modified corpus-based approach + genetic algorithm (MCBA + GA) of Yeh et al. (2005)

We are going to exploit the MCBA + GA approach of Yeh et al., for summarization and use it as a baseline approach.

For a sentence s, a weighted score function, as shown in the following equation is exploited to integrate the first 5 feature scores mentioned in Section 2, where w_i indicates the weight of f_i

$$Score(s) = w_1 \cdot Score_{f_1}(s) + w_2 \cdot Score_{f_2}(s) + w_3 \cdot Score_{f_3}(s) + w_4 \cdot Score_{f_4}(s) + w_5 \cdot Score_{f_5}(s) \tag{32}$$

We use the approach as described in Section 3.1.

For testing, a set of 100 Arabic and 100 English documents was used. We apply Eq. (32) after using the defined weights from GA execution. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate. Tables 1 and 2 show the MCBA + GA approach performance evaluation based on precision using the first five features for Arabic and English documents, respectively.

4.3. The effect of each feature on summarization performance

In this section, we investigate the effect of each feature parameter on summarization by using Eq. (10) with individual score using feature weight equal to 1. For instance, to investigate the first feature (sentence position) on summarization performance, we use the following equation:

$$Score(s) = Score_{f_1}(s) \tag{33}$$

Moreover, it is worth to conduct some experiments based on the simple lead approach. The lead method is known to be effective for document summarization of newspapers in lower compression ratio.

Tables 3 and 4 show the summarization precision associated with lead approach and each feature for different compression rates for Arabic and English documents respectively.

Table 1
The MCBA + GA approach performance evaluation based on precision (Arabic case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4239	0.4412	0.4321
95% Confidence interval	0.4076, 0.4403	0.4164, 0.466	0.4113, 0.453

Table 2
The MCBA+GA approach performance evaluation based on precision (English case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4253	0.4265	0.4278
95% Confidence interval	0.409, 0.4417	0.4057, 0.4474	0.4056, 0.45

Table 3
The summarization precision associated with each feature for different compression rates (Arabic case)

Compression rate (CR)	10%	95% Confidence interval	20%	95% Confidence interval	30%	95% Confidence interval
Lead approach	0.3012	0.2764, 0.3260	0.2752	0.2589, 0.2916	0.2510	0.2347, 0.2674
P(f1)	0.3523	0.3315, 0.3732	0.3625	0.3417, 0.3834	0.3536	0.3328, 0.3745
P(f2)	0.3428	0.3206, 0.3650	0.3414	0.3192, 0.3636	0.3342	0.3120, 0.3564
P(f3)	0.3023	0.2834, 0.3213	0.2923	0.2734, 0.3113	0.2987	0.2798, 0.3177
P(f4)	0.4023	0.3775, 0.4271	0.4045	0.3882, 0.4209	0.4086	0.3923, 0.4250
P(f5)	0.3623	0.3413, 0.3834	0.3723	0.3513, 0.3934	0.3873	0.3663, 0.4084
P(f6)	0.3243	0.3114, 0.3371	0.3134	0.3005, 0.3262	0.3256	0.3127, 0.3384
P(f7)	0.2645	0.2423, 0.2867	0.2635	0.2413, 0.2857	0.2535	0.2313, 0.2757
P(f8)	0.2847	0.2658, 0.3037	0.2798	0.2609, 0.2988	0.2865	0.2676, 0.3055
P(f9)	0.4265	0.4017, 0.4513	0.4236	0.3988, 0.4484	0.4137	0.3889, 0.4385
P(f10)	0.3825	0.3615, 0.4036	0.3634	0.3424, 0.3845	0.3736	0.3526, 0.3947

Table 4
The summarization precision associated with each feature for different compression rates (English case)

Compression rate (CR)	10%	95% Confidence interval	20%	95% Confidence interval	30%	95% Confidence interval
Lead approach	0.2843	0.2670, 0.3017	0.2532	0.2359, 0.2706	0.2216	0.2043, 0.2390
P(f1)	0.3365	0.3127, 0.3604	0.3487	0.3249, 0.3726	0.3424	0.3186, 0.3663
P(f2)	0.3332	0.3090, 0.3574	0.3318	0.3076, 0.3560	0.3456	0.3214, 0.3698
P(f3)	0.2897	0.2718, 0.3076	0.2975	0.2796, 0.3154	0.2998	0.2819, 0.3177
P(f4)	0.3854	0.3681, 0.4028	0.3993	0.3820, 0.4167	0.4097	0.3924, 0.4271
P(f5)	0.3587	0.3337, 0.3838	0.3548	0.3298, 0.3799	0.3790	0.3540, 0.4041
P(f6)	0.3193	0.3014, 0.3371	0.3285	0.3106, 0.3463	0.3264	0.3085, 0.3442
P(f7)	0.2612	0.2330, 0.2894	0.2698	0.2416, 0.2980	0.2665	0.2383, 0.2947
P(f8)	0.2698	0.2504, 0.2893	0.2652	0.2458, 0.2847	0.2747	0.2553, 0.2942
P(f9)	0.4153	0.3915, 0.4391	0.4176	0.3938, 0.4414	0.4283	0.4045, 0.4521
P(f10)	0.3749	0.3518, 0.3981	0.3682	0.3451, 0.3914	0.3794	0.3563, 0.4026

4.4. The results of genetic algorithm model (GA)

We have exploited the MCBA + GA approach of Yeh et al., for summarization as described in Section 4.2 using Eq. (10) rather than Eq. (32). Therefore, we have exploited the 10 features for summarization. The system calculates the feature weights using Genetic algorithm.

All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate. Tables 5 and 6 show the GA approach performance evaluation based on precision using the 10 features for Arabic and English documents, respectively.

4.5. The results of mathematical regression (MR)

We have exploited the approach in Section 3.2. For testing, a set of 100 Arabic and 100 English documents was used. We applied Eq. (10) after using the defined weights from [*W*]. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate. Tables 7 and 8 show the MR approach performance evaluation based on precision for different compression rates for Arabic and English documents, respectively.

Table 5
The GA approach performance evaluation based on precision (Arabic case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4486	0.4591	0.4563
95% Confidence interval	0.4307, 0.4665	0.4418, 0.4765	0.4369, 0.4758

Table 6
The GA approach performance evaluation based on precision (English case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4376	0.4435	0.4494
95% Confidence interval	0.4197, 0.4555	0.4262, 0.4609	0.43, 0.4689

Table 7
The MR approach performance evaluation based on precision (Arabic case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4489	0.4496	0.4475
95% Confidence interval	0.4310, 0.4668	0.4323, 0.4670	0.4281, 0.4670

Table 8
The MR approach performance evaluation based on precision (English case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4312	0.4353	0.4382
95% Confidence interval	0.4123, 0.4501	0.417, 0.4537	0.4168, 0.4597

4.6. *The results of feed forward neural network (FFNN)*

The system is extracting features from the 100 Arabic and 50 English manually summarized documents and uses them to train the feed forward back propagation neural network. Use the other 100 Arabic and 100 English documents as a testing set. Then apply the sentences of these documents as inputs to the feed forward back propagation neural network after feature extraction step as follows:

1. Extract features from the sentences of the document.
2. Construct the feature vector $\overrightarrow{X}$ as shown in Fig. 3.
3. Use this feature vector as an input of the neural network.
4. Save the output of the neural network for each sentence (the output is a value between 0 and 1).
5. Rank all document sentences based on their scores then arrange them in a descending order.
6. Chronologically select the set of sentences of highest scores based on the required compression rate.

Tables 9 and 10 show the results of FFNN for the 100 Arabic and 100 English articles respectively.

4.7. *The results of probabilistic neural network (PNN)*

The system is extracting features from the 100 Arabic and 50 English manually summarized documents and uses them to train probabilistic neural network. Use the other 100 Arabic and 100 English documents as a testing set. Then apply the sentences of these documents as inputs to the Probabilistic Neural Network after feature extraction step as follows:

1. Extract features from the sentences of the document.
2. Construct the feature vector X as shown in Fig. 4.
3. Use this feature vector as an input of the probabilistic neural network.
4. Save the output of the neural network for each sentence (the result of formula (27).
5. Rank all document sentences based on their scores then arrange them in a descending order.
6. Chronologically select the set of sentences of highest scores based on the required compression rate.

Tables 11 and 12 show the results of PNN for the 100 Arabic and 100 English articles, respectively.

Table 9
The FFNN approach performance evaluation based on precision (Arabic case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4686	0.4593	0.4665
95% Confidence interval	0.4497, 0.4875	0.441, 0.4777	0.4451, 0.488

Table 10
The FFNN approach performance evaluation based on precision (English case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4654	0.4632	0.4695
95% Confidence interval	0.4465, 0.4843	0.4449, 0.4816	0.4481, 0.4910

Table 11
The PNN approach performance evaluation based on precision (Arabic case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4592	0.4665	0.4712
95% Confidence interval	0.4423, 0.4761	0.4502, 0.4829	0.4508, 0.4917

Table 12
The PNN approach performance evaluation based on precision (English case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4532	0.4597	0.4701
95% Confidence interval	0.4363, 0.4701	0.4434, 0.4761	0.4497, 0.4906

4.8. The results of Gaussian mixture model (GMM)

The system is extracting features from the sentences of the 100 Arabic and 50 English manually summarized documents and uses them to construct Gaussian mixture model for each category (we have two categories). Use the other 100 Arabic and 100 English documents as a testing set. Then apply the sentences of these documents as inputs to the Gaussian mixture model after feature extraction step as follows:

1. Extract features from the sentences of the document.
2. Construct the feature vector X.
3. Use this feature vector as an input of the GMM.
4. Save the output of the GMM for each sentence (the result of formula (30)).
5. Rank all document sentences based on their scores then arrange them in a descending order.
6. Chronologically select the set of sentences of highest scores based on the required compression rate.

Tables 13 and 14 show the results of GMM for the 100 Arabic and 100 English articles, respectively.

4.9. The results of all models trained on Arabic data and tested on English data

In this experiment, we train all previously mentioned models on 8 Arabic features (we exclude f2 = positive keyword and f3 = negative key word since they are language dependent) and test these models on English data to check whether these eight features are language dependent or not. Table 15 shows the results of all models for the 100 English articles.

Table 13
The GMM approach performance evaluation based on precision (Arabic case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.6152	0.5985	0.6276
95% Confidence interval	0.5983, 0.6321	0.5822, 0.6149	0.6072, 0.6481

Table 14
The GMM approach performance evaluation based on precision (English case)

Compression rate (CR)	10%	20%	30%
Precision (P)	0.6025	0.6076	0.6178
95% Confidence interval	0.5856, 0.6194	0.5913, 0.6240	0.5974, 0.6383

Table 15
All models performance evaluation based on precision (English testing data)

Compression rate (CR)	10%	95% Confidence interval	20%	95% Confidence interval	30%	95% Confidence interval
P(GA)	0.4243	0.4070, 0.4417	0.4321	0.4148, 0.4495	0.4376	0.4203, 0.4550
P(MR)	0.4153	0.3915, 0.4392	0.4038	0.3800, 0.4277	0.4063	0.3825, 0.4302
P(FFNN)	0.4464	0.4222, 0.4706	0.4412	0.4170, 0.4654	0.4453	0.4211, 0.4695
P(PNN)	0.4453	0.4274, 0.4632	0.4586	0.4407, 0.4765	0.4603	0.4424, 0.4782
P(GMM)	0.5923	0.5750, 0.6097	0.5976	0.5803, 0.615	0.6092	0.5919, 0.6266

4.10. The results of all models trained on English data and tested on Arabic data

In this experiment, we train all previously mentioned models on eight English features (we exclude f2 = positive keyword and f3 = negative key word since they are language dependent) and test these models on Arabic data like the previously mentioned experiment. Table 16 shows the results of all models for the 100 Arabic articles.

4.11. The results of all models trained on English data and tested on DUC 2001 data

In this experiment, we train all previously mentioned models on the 10 English features (using the same 50 English articles) and test these models on the DUC 2001 data to investigate the proposed system performance on a newswire data. We have created new extractive reference summaries of the DUC 2001 testing data by measuring the similarity (vocabulary overlap) between each sentence and the associated reference single document summary. Then we rank each document sentences based on this similarity value. A set of sentences is specified as a reference summary for each document based on the compression ratio. Table 17 shows the results of all models for the DUC 2001 testing data based on precision. Table 18 shows the results of all models for the DUC 2001 testing data based on the average Rouge-1 score.

4.12. Discussion

It is clear from Tables 3 and 4 that the most important text feature for summarization is f9 (Bushy path) since it gives the best results. It is reasonable, since the sentence that has a maximum number of branches should convey the most important part in the article. f4 (centrality) also gives good results since it conveys

Table 16
All models performance evaluation based on precision (Arabic testing data)

Compression rate (CR)	10%	95% Confidence interval	20%	95% Confidence interval	30%	95% Confidence interval
P(GA)	0.4076	0.3893, 0.4260	0.4193	0.4010, 0.4377	0.4232	0.4049, 0.4416
P(MR)	0.4085	0.3857, 0.4314	0.4054	0.3826, 0.4283	0.3985	0.3757, 0.4214
P(FFNN)	0.4476	0.4264, 0.4688	0.4487	0.4275, 0.4699	0.4434	0.4222, 0.4646
P(PNN)	0.4414	0.4225, 0.4603	0.4486	0.4297, 0.4675	0.4676	0.4487, 0.4865
P(GMM)	0.5887	0.5704, 0.6071	0.5954	0.5771, 0.6138	0.6036	0.5853, 0.6220

Table 17
All models performance evaluation based on precision (DUC 2001 testing data)

Compression rate (CR)	10%	95% Confidence interval	20%	95% Confidence interval	30%	95% Confidence interval
P(GA)	0.4152	0.3968, 0.4335	0.4236	0.4053, 0.4419	0.4335	0.4132, 0.4538
P(MR)	0.4098	0.3869, 0.4326	0.4021	0.3793, 0.4249	0.4021	0.3803, 0.4239
P(FFNN)	0.4383	0.4171, 0.4595	0.4403	0.4191, 0.4615	0.4423	0.4191, 0.4655
P(PNN)	0.4438	0.4249, 0.4627	0.4526	0.4337, 0.4715	0.4543	0.4344, 0.4742
P(GMM)	0.5902	0.5718, 0.6085	0.5936	0.5753, 0.6119	0.6046	0.5873, 0.6219

Table 18
All models performance evaluation based on the average Rouge-1 score (DUC 2001 testing data)

Compression rate (CR)	10%	95% Confidence interval	20%	95% Confidence interval	30%	95% Confidence interval
Av_Rouge-1(GA)	0.4326	0.4142, 0.4509	0.4435	0.4251, 0.4618	0.4534	0.4350, 0.4717
Av_Rouge-1(MR)	0.4302	0.4106, 0.4497	0.4314	0.4118, 0.4509	0.4357	0.4161, 0.4552
Av_Rouge-1(FFNN)	0.4543	0.4309, 0.4777	0.4626	0.4392, 0.4860	0.4678	0.4444, 0.4912
Av_Rouge-1(PNN)	0.4587	0.4330, 0.4844	0.4756	0.4499, 0.5013	0.4793	0.4536, 0.5050
Av_Rouge-1(GMM)	0.6075	0.5778, 0.6372	0.6124	0.5827, 0.6421	0.6257	0.5960, 0.6554

the vocabulary overlap between this sentence and other sentences in the document. Usually, the document title conveys the main topic of this document. Therefore, f5 (sentence resemblance to the title) which is the vocabulary overlap between this sentence and the document title gives good results. The lowest results are associated with f7 (sentence inclusion of numerical data) since most of political and religious articles do not contain many numerical data. Therefore, the system ranks a sentence that does not contain numerical data according to its position.

It is clear from Tables 15 and 16 that the precisions have decreased slightly when the models are trained on one language and tested on the other language. However, the decrease in precision was not significant. There-

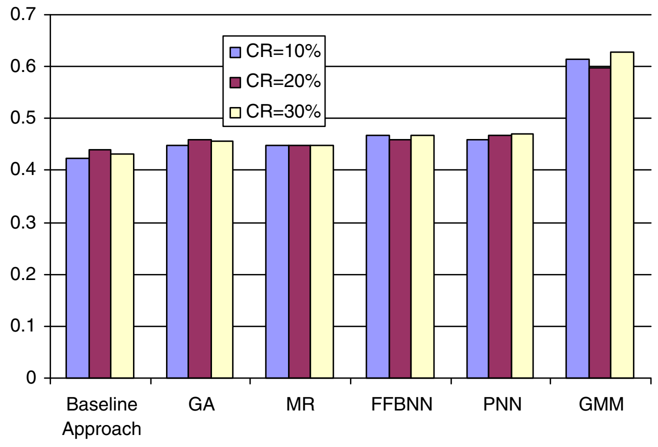

Fig. 5. The results associated with all models for different CR (Arabic case).

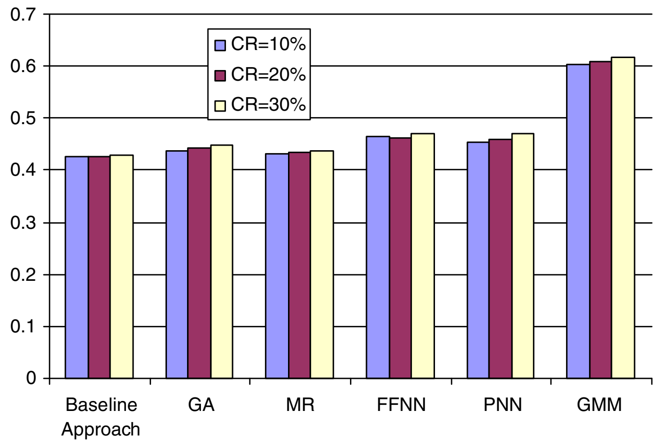

Fig. 6. The results associated with all models for different CR (English case).

fore, it is possible to train some models with some features and use them for another language. Moreover, it is clear from Table 17 that this approach can be extended to the genre of newswire text.

Figs. 5 and 6 show the total system performance in terms of precision for different compression rates in case of all models for Arabic and English articles, respectively. It is clear from the figures that GMM approach gives the best results since GMM has a good capability to model arbitrary densities. The PNN approach has better precision than the FNNN approach then GA approach then MR approach. It is also clear that decreasing the training data size to half (English case) does not severely affect the total system performance.

5. Conclusions and future work

In this paper, we have investigated the use of genetic algorithm (GA), mathematical regression (MR), feed forward neural network (FFNN), probabilistic neural network (PNN) and Gaussian mixture model (GMM) for automatic text summarization task. We have applied our new approaches on a sample of 100 Arabic political articles and 100 English religious articles. Our approach results outperform the baseline approach results. Our approaches have been used the feature extraction criteria which gives researchers opportunity to use many varieties of these features based on the used language and the text type. Some text features are language dependent like positive and negative keywords while some other features are language independent. We can train the models on a data of a certain language and test these models on a data of another language.

In the future work, we will extend this approach to multi-document summarization by addressing some anti-redundancy methods which are needed, since the degree of redundancy is significantly higher in a group of topically related articles than in an individual article as each article tends to describe the main point as well as necessary shared background.

Acknowledgment

This research has been partially supported by the Japan Society for the Promotion of Science (JSPS), Grant No. 07077, and the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (B), 19300029.

References

Aone, C., Okurowski, M.E., Gorlinsky, J., Larsen, B., 1997. A scalable summarization system using robust NLP. In: Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization, Madrid, Spain, pp. 10–17.

Azzam, S., Humphreys, K., Gaizauskas, R., 1999. Using coreference chains for text summarization. In: Proceedings of the ACL'99, College Park, MD, USA, pp. 77–84.

Cain, B.J., 1990. Improved probabilistic neural networks and its performance relative to the other models. Proceedings of SPIE, Applications of Artificial Neural Networks 1294, 354–365.

Carbonell, J.G., Goldstein, J., 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st ACM SIGIR, pp. 335–336.

Conroy, J., O'Leary, J., 2001. Text summarization via hidden Markov models. In: SIGIR, pp. 406–407.

Diaz, A., Gervás, P., 2007. User-model based personalized summarization. Information Processing & Management 43 (6), 1715–1734.

Dorr, B., Gaasterland, T., 2007. Exploiting aspectual features and connecting words for summarization-inspired temporal-relation extraction. Information Processing & Management 43 (6), 1681–1704.

Edmundson, H.P., 1969. New methods in automatic extracting. Journal of the ACM (JACM) 16 (2), 264–285.

Erkan, G., Radev, D., 2004. LexRank: graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research 22, 457–479.

Fattah, M., Ren, F., Kuroiwa, S., 2006a. Effects of phoneme type and frequency on distributed speaker identification and verification. The International Journal "IEICE" E89-D (5), 1712–1719.

Fattah, M., Ren, F., Kuroiwa, S., 2006b. Sentence alignment using feed forward neural network. International Journal of Neural Systems (IJNS) 16 (6), 423–434.

Ganchev, T., Tasoulis, D.K., Vrahatis, M.N., Fakotakis, N., 2003. Locally recurrent probabilistic neural networks for text independent speaker verification. In: Proceedings of the EuroSpeech, vol. 3, pp. 1673–1676.

Goldstein, J., Kantrowitz, M., Mittal, V., Carbonell, J., 1999. Summarizing text documents: sentence selection and evaluation metrics. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99), Berkeley, CA, USA, pp. 121–128.

Gong, Y., Liu, X., 2001. Generic text summarization using relevance measure and latent semantic analysis. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01), New Orleans, LA, USA, pp. 19–25.

Hahn, U., Mani, I., 2000. The challenges of automatic summarization. IEEE-Computer 33 (11), 29–36.

Harabagiu, S., Hickl, A., Lacatusu, F., 2007. Satisfying information needs with multi-document summaries. Information Processing & Management 43 (6), 1619–1642.

Hirao, T., Okumura, M., Yasuda, N., Isozaki, H., 2007. Supervised automatic evaluation for summarization with voted regression model. Information Processing & Management 43 (6), 1521–1535.

Hobson, S., Dorr, B., Monz, C., Schwartz, R., 2007. Task-based evaluation of textsummarization using relevance prediction. Information Processing & Management 43 (6), 1482–1499.

Hovy, E., Lin, C.Y., 1997. Automatic text summarization in SUMMARIST. In: Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization, Madrid, Spain, pp. 18–24.

Jann, B., 2005. Making regression tables from stored estimates. Stata Journal 5, 288–308.

Jing, B.H., 2002. Using hidden markov modeling to decompose Human-Written summaries. Computational Linguistics 28 (4), 527–543.

Kleinberg, J., Tardos, E., 2002. Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. Journal of the Association for Computing Machinery 49 (5), 616–639.

Kupiec, J., Pedersen, J., Chen, F., 1995. A trainable document summarizer. In: Proceedings of the 18thAnnual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95), Seattle, WA, USA, pp. 68–73.

Ling, X., Jiang, J., He, X., Mei, Q., Zhai, C., Schatz, B., 2007. Generating gene summaries from biomedical literature: a study of semi-structured summarization. Information Processing & Management 43 (6), 1777–1791.

Luhn, H.P., 1958. The automatic creation of literature abstracts. IBM Journal of Research and Development 2 (2), 159–165.

Mani, I., Bloedorn, E., 1999. Summarizing similarities and differences among related documents. Information Retrieval 1 (1–2), 35–67.

Mani, I., Maybury, M.T. (Eds.), 1999. Advances in Automated Text Summarization. The MIT Press, Cambridge, MA.

McKeown, K., Radev, D.R., 1995. Generating summaries of multiple news articles. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95), Seattle, WA, USA, pp. 74–82.

Moens, M., 2007. Summarizing court decisions. Information Processing & Management 43 (6), 1748–1764.

Nomoto, T., 2007. Discriminative sentence compression with conditional random fields. Information Processing & Management 43 (6), 1571–1587.

Nomoto, T., Matsumoto, Y., 2001. A new approach to unsupervised text summarization. In: Proceedings of the 24th ACM SIGIR, pp. 26-34.

Over, P., Dang, H., Harman, D., 2007. DUC in context. Information Processing & Management 43 (6), 1506–1520.

Pellom, B.L., Hansen, J.H.L., 1998. An efficient scoring algorithm for gaussian mixture model based speaker identification. IEEE Signal Processing Letters 5 (11), 281–284.

Reeve, L., Han, H., Brooks, A., 2007. The use of domain-specific concepts in biomedical text summarization. Information Processing & Management 43 (6), 1765–1776.

Reynolds, D., 1995. Speaker identification and verification using Gaussian mixture speaker models. Speech Communication 17, 91–108.

Richard, W., 2006. Review of regression models forcategorical dependent variables using Stata, Second Edition, by Long and Freese. The Stata Journal 6 (2), 273–278.

Russell, S.J., Norvig, P., 1995. Artificial Intelligence: A Modern Approach. Prentice-Hall International Inc., Englewood Cliffs, NJ.

Salton, G., Singhal, A., Mitra, M., Buckley, C., 1997. Automatic text structuring and summarization. Information Processing & Management 33 (2), 193–207.

Schank, R., Abelson, R., 1977. Scripts, Plans, Goals, and Understanding. Lawrence Erlbaum Associates, Hillsdale, NJ.

Sjöbergh, J., 2007. Older versions of the ROUGEeval summarization evaluation system were easier to fool. Information Processing & Management 43 (6), 1500–1505.

Sparck Jones K., 1993. Discourse modeling for automatic summarizing. Technical Report 29D, Computer Laboratory, University of Cambridge.

Specht, D.F., 1990. Probabilistic neural networks. Neural Networks 3 (1), 109–118.

Steinberger, J., Poesio, M., Kabadjov, M., Ježek, K., 2007. Two uses of anaphora resolution in summarization. Information Processing & Management 43 (6), 1663–1680.

Teufel, S.H., Moens, M., 1997. Sentence extraction as a classification task. In: Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization, Madrid, Spain, pp. 58–65.

Vanderwende, L., Suzuki, H., Brockett, C., Nenkova, A., 2007. Beyond SumBasic: task-focused summarization with sentence simplification and lexical expansion. Information Processing & Management 43 (6), 1606–1618.

Ye, J.S., Chua, H.T., Kan, W.M., Qiu, I.L., ml_chg_old>Yeh Ye et al., 2007. Document concept lattice for text understanding and summarization. Information Processing & Management 43 (6), 1643–1662.

Yeh, J.Y., Ke, H.R., Yang, W.P., 2002. Chinese text summarization using a trainable summarizer and latent semantic analysis. In: Proceedings of the 5th International Conference on Asian Digital Libraries (ICADL'02)Lecture Notes in Computer Science, vol. 2555. Springer-Verlag, Singapore, Berlin, pp. 76–87.

Yeh, S.J., Ke, T.H., Yang, M.W., Meng, L.I., ml_chg_old>Ye Yeh et al., 2005. Text summarization using a trainable summarizer and latent semantic analysis. Information Processing & Management 41 (1), 75–95.

Young, S.R., Hayes, P.J., 1985. Automatic classification and summarization of banking telexes. In: Proceedings of the 2nd Conference on Artificial Intelligence Application, pp. 402–408.

Zajic, D., Dorr, B., Lin, J., Schwartz, R., 2007. Multi-candidate reduction: sentence compression as a tool for document summarization tasks. Information Processing & Management 43 (6), 1549–1570.

Zhu, X., Ghahramani, Z., 2002. Towards semi-supervised classification with Markov random fields. Technical Report, CMU-CALD-02-106.