Performance Evaluation for Question Classification by Tree Kernels using Support Vector Machines

Muhammad Arifur Rahman
Department of Physics, Jahangirnagar University
Dhaka-1342, Bangladesh
arifmail@gmail.com

Abstract — **Question answering systems use information retrieval (IR) and information extraction (IE) methods to retrieve documents containing a valid answer. Question classification plays an important role in the question answer frame to reduce the gap between question and answer. This paper presents our research work on automatic question classification through machine learning approaches. We have experimented with machine learning algorithms Support Vector Machines (SVM) using kernel methods. An effective way to integrate syntactic structures for question classification in machine learning algorithms is the use of tree kernel (TK) functions. Here we use SubTree kernel, SubSet Tree kernel with Bag of words and Partial Tree kernels. Trade-off between training error and margin, Cost-factor and the decay factor has significant impact when we use SVM for the above mentioned kernel types. The experiments determined the individual impact for Trade-off between training error and margin, Cost-factor and the decay factor and later the combined effect for Trade-off between training error and margin, Cost-factor. For each kernel types depending on these result we also figure out some hyper planes which can maximize the performance. Based on some standard data set outcomes of our experiment for question classification is promising.**

Index Terms — **Precision, Recall, kernel, SubSet Tree, SubTree, Partial Tree, SVM, Question Classification, Question Answering.**

I. INTRODUCTION

The World Wide Web continues to grow at an amazing speed. So, there are also a quickly growing number of text and hypertext documents. Due to the huge size, high dynamics, and large diversity of the web, it has become a very challenging task to find the truly relevant content for some user or purpose. The open-domain question answering system (QA) has been attached great attention for its capacity of providing compact and precise results for users.

The study of question classification (QC), as a new field, corresponds with the research of QA. At present the studies on QC are mainly based on the text classification. Though QC is similar to text classification in some aspects, they are clearly distinct in that: Question is usually shorter, and contains less lexicon-based information than text, which brings great trouble to QC. Therefore to obtain higher classifying accuracy, QC has to make further analysis of sentences, namely QC has to extend interrogative sentence with syntactic and semantic knowledge, replacing or extending the vocabulary of the question with the semantic meaning of every word.

In QC, many systems apply machine-learning approaches [1-3]. The classification is made according to the lexical, syntactic features and parts of speech. Machine learning approach is of great adaptability, and 90.0% of classifying accuracy is obtained with SVM method and tree kernel as features. However, there is still the problem that the classifying result is affected by the accuracy of syntactic analyzer, which need manually to determine the weights of different classifying features. Some other systems adopting manual-rule [4].

This paper presents our approaches at question classification to improve the f1-measure [6]. of question categorization. Our experiment tried to determine the optimum value for trade-off between training error and margin, cost-factor by which training errors on positive examples outweighs errors on negative examples. It is used to adjust the rate between precision and recall on the development set [5-6]. and the decay factor which has the role to penalize larger tree structures by giving them less weight in the overall summation [7]. To represent questions, the paper uses the standard Li and Roth [8] question classification dataset.

After this overview, question classifying method, a brief description about the used algorithm and the kernel methods are introduced, and then impact of different parameters and parameters combination methods has been investigated. The comparisons are testified in experiments based on precision, recall and f1-measure. The last part of the paper is about the conclusion of the present research and about the introduction of the further work could be done on this issue.

II. QUESTION CLASSIFICATION

Question classification means putting the questions into several semantic categories. Approaches to question classification can be divided in two broad classes, namely, rule-based and machine learning methods. Most recent studies have been based on machine learning approaches. Li and Roth proposed 6 coarse classes and 50 fine classes for TREC factoid question answering. The UIUC QC dataset, which they developed, contains 5,500 training questions and 500 test questions, and it is now the standard dataset for question classification [8-9]. We used this dataset for our training and testing purpose.

In Table I each coarse grained category contains a non-overlapping set of fine grained categories. Most question answering systems use a coarse grained category definition. Usually the number of question categories is less than 20. However, it is obvious that a fine grained category definition is more beneficial in locating and verifying the plausible answers.

TABLE I: THE COARSE AND FINE GRAINED QUESTION CATEGORIES.

Coarse	Fine
ABBR	Abbreviation, expansion
DESC	description, definition, manner, reason
ENTY	body, color, event, food, creation, currency, animal, disease/medical, instrument, language, letter, other, plant, product, religion, sport, substance, symbol, technique, term, vehicle, word
HUM	description, group, individual, title
LOC	city, country, mountain, other, state
NUM	code, count, date, distance, money, order, other, percent, period, speed, temperature, size, weight

III. SUPPORT VECTOR MACHINE (SVM)

Support vector machine are based on the structural risk minimization principle from statistical learning theory [10]. The idea of structural risk minimization is to find a hypothesis for which we can guarantee the lowest true error. The bounds are connected on the true error with the margin of separating hyper planes. In their basic form support vector machines find the hyper plane that separates the training data with maximum margin. Since we will be dealing with very unbalanced numbers of positive and negative examples in the following, we use cost factors C_+ and C_- to be able to adjust the cost of false positives vs. false negatives as [11]. Finding this hyper plane can be translated into the following optimization problem:

$$\text{Minimize: } \frac{1}{2}\|\vec{w}\|^2 + C_+ \sum_{i:y_i=1} \xi_i + C_- \sum_{i:y_j=-1} \xi_j$$

$$\text{Subjected to: } \forall k : y_k\left[\vec{w}.\vec{x}_k + b \geq 1 - \xi_k\right]$$

Here x_i is the feature vector of example i. y_i equals +1 or -1, if example i is in class '+' or '-' respectively.

Using support vector machine we tried to determine the optimal value of trade-off between training error and margin (c), cost-factor (j) by which training errors on positive examples outweighs errors on negative examples. It is used to adjust the rate between precision and recall on the development set [11]. Here lambda (λ) the decay factor which has the role to penalize larger tree structures by giving them less weight in the overall summation.

IV. KERNEL METHODS

One of the most difficult tasks on applying machine learning for question classification is the feature design. Feature should represent data in a way that allows learning algorithm to separate positive from negative examples. In SVMs, features are used to build the vector representation of data examples and the scalar product between example pairs quantifiers how much they are similar. Instead of encoding data in the features vectors, kernel functions can be designed that provide such similarity between example pairs without using an explicit feature representation [6].

The kernels we considered in this paper represent trees in terms of their substructure. Such fragments define the feature space which, in turn, is mapped into a vector space. The kernel function measures the similarity between trees by counting the number of common fragments. These functions have to recognize if a common tree subpart belongs to the feature space that we intended to generate. Here we considered three important characterization: SubTrees (STs), SubSet Trees (SSTs) and a new tree class Partial Trees (PTs) described as in [6].

In case of syntactic parse trees each node with its children is associated with a grammar production rule, where the symbol at left hand side corresponds to the parent and the symbol at right hand side are associated with the children. The terminal symbols of the grammar are always associated with the leaves of the tree. For example, Fig 1. illustrate the syntactic parse of the sentence-

`"Adib brought a cat to the school".`

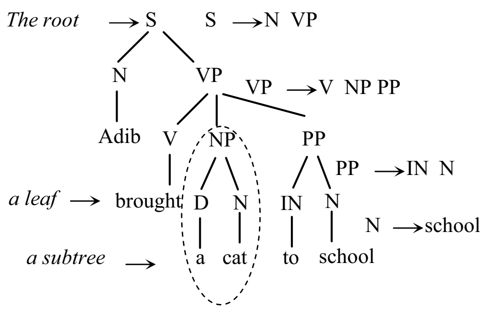

Figure. 1. A syntactic parse tree.

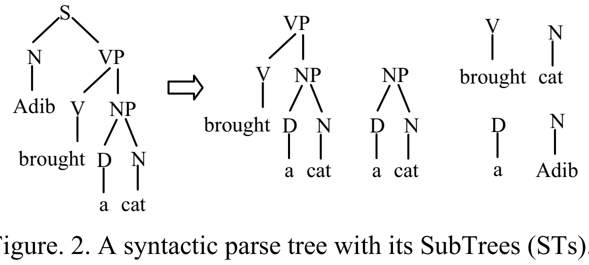

Figure. 2. A syntactic parse tree with its SubTrees (STs).

A SubTree (ST) can be defined as any node of a tree along with all its descendants. For example, in the Fig. 1. circle the SubTree rooted in the NP node. A SubSet Tree (SST) is a more general structure which necessarily not includes all the descendants. The only restriction is that an SST must be generated by applying the same grammatical rule set which generated the original tree, as pointed out in [12]. Thus, the difference with the SubTree is that the SST's leaves can be associated with non-terminal symbols. For example, [S [N VP]] is an SST of the tree in the Fig. 1. and it has the two non terminal symbols N and VP as leaves.

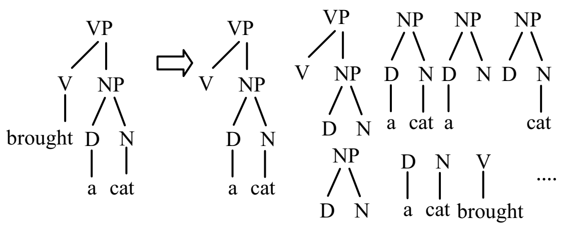

Figure. 3. A tree with some of its SubSet Trees (SSTs).

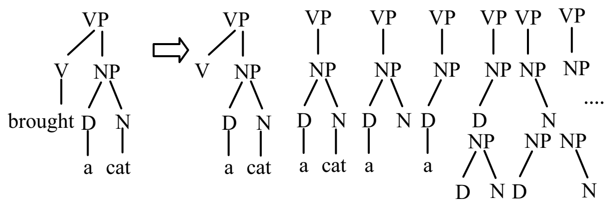

Figure. 4. A tree with some of its Partial Trees (PTs).

If we relax the constraints over the SST's, we obtain a more general form of substructure that we defined as Partial Trees (PTs). These can be generated by the application of partial production rules of the original grammar. For example, [S [N VP]], [S [N]] and [S [VP]] are valid PTs of the tree in the Fig. 1. Here Fig. 2. shows the parse tree of the sentence "`Adib brought a cat`" together with its 6 STs. The number of SSTs is always higher. For example, Fig. 3. shows 10 SSTs (out of all 17) of the SubTree of the Fig. 2. rooted in VP. Fig. 4. shows that the number of PTs derived from the same tree is still higher (i.e. 30 PTs). These different substructure numbers provide an intuitive quantification of the different information level among the tree based representations [6].

The main idea of the tree kernel is to compute the number of the common substructures between two trees T_1 and T_2 without explicitly considering the whole fragment space. From [6]. given a tree fragment space $\{f_1, f_2, \ldots\} = F$, here we denote the indicator function $I_i(n)$ which is equal 1 if the target f_1 is rooted at the node n and 0 otherwise. It followed that-

$$K(T_1,T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1,n_2) \ldots \quad (1)$$

where N_{T_1} and N_{T_2} are the sets of the T_1's and T_2's nodes, respectively and $\Delta(n_1,n_2) = \sum_{i=1}^{F} I_i(n_1) I_i(n_2)$.

This letter is equal to the number of common fragments rooted at the n_1 and n_2 nodes. We can compute Δ as follows-

1. if the production at n_1 and n_2 are different the $\Delta(n_1,n_2) = 0$;
2. if the production at n_1 and n_2 are the same, and n_1 and n_2 have only leaf children (pre-terminal symbols) then $\Delta(n_1,n_2) = 1$;
3. if the production at n_1 and n_2 are the same, and n_1 and n_2 are not pre-terminals then

$$\Delta(n_1,n_2) = \prod_{j=1}^{nc(n_1)} \left(\sigma + \Delta\left(c_{n_1}^{j}, c_{n_2}^{j}\right)\right) \ldots \quad (2)$$

where $\sigma \in \{0,1\}$, $nc(n_1)$ is the number of the children of n_1 and c_n^j is the j-th child of the node n. As the production is same $nc(n_1) = nc(n_2)$.

When $\sigma = 0$, $\Delta(n_1,n_2)$ is equal 1 only if $\forall j \, \Delta\left(c_{n_1}^{j}, c_{n_2}^{j}\right) = 1$, i.e. all the productions associated with the children are identical. By recursively applying the property, it follows the SubTrees in n_1 and n_2 are identical. Thus Eq. 1 evaluate the SubSet Tree kernel .When $\sigma = 1$, $\Delta(n_1,n_2)$ evaluates the number of SSTs common to n_1 and n_2 as proved in [12].

To include the leaves as fragments it is enough to add, to the recursive rule set for the Δ evaluation, the condition is- if n_1 and n_2 are leaves and their associated symbols are equal then $\Delta(n_1,n_2) = 1$. By [6]. such extended kernel can be referred as SST+bow (*bag-of-words*). Here the decay factor (λ) is defined as $\Delta(n_x,n_z) = \lambda$ and

$$\Delta(n_x,n_z) = \lambda \prod_{j=1}^{nc(n_x)} \left(\sigma + \Delta\left(c_{n_1}^{j}, c_{n_2}^{j}\right)\right).$$

To evaluate all possible substructures common to two trees, we can (1) select a child from the both trees, (2) extract the portion of the syntactic rule that contains such subset, (3) apply Eq. 2 to the extracted partial productions and sum the contributions of all the children subsets. Such subsets corresponds to all possible common (non continuous) node subsequence and computed efficiently by means of sequence kernel [13]. Let $\vec{J}_1 = (J_{11}, .., J_{1r})$ and $\vec{J}_2 = (J_{21}, .., J_{2r})$ be the index sequence associated with the order child sequence of n_1 and n_2 respectively. Then the number of PTs is evaluated by the following Δ function:

$$\Delta(n_1,n_2) = 1 + \sum_{\vec{J}_1, \vec{J}_2, l(\vec{J}_1) = l(\vec{J}_2)} \prod_{i=1}^{l(\vec{J}_1)} \Delta\left(c_{n_1}^{\vec{J}_{1i}}, c_{n_2}^{\vec{J}_{2i}}\right)$$

where $l(\vec{J}_1)$ indicates the length of the target child sequence, whereas $\vec{J}_{1i}$ and $\vec{J}_{2i}$ are the i^{th} children in the two sequences.

V. RESULTS AND ANALYSIS

We evaluate the performance of Question Classification using SVM with SubTree kernel, SubSet Tree kernel with Bag of words (SST+BOW) and Partial Tree kernel. For SubSet Tree kernel with bag of words first we investigate the optimum value of trade-off between training error and margin (c) keeping cost-factor (j) and the decay factor (λ) constant. Then we investigate the impact of cost factor (j) for constant trade-off between training error and margin (c) and decay factor (λ). Then for constant factor (λ) we determine the hyper plane for trade-off between training error and margin (c) and cost-factor (j) which have the maximum f1-measure. Finally we determined the impact of decay factor (λ) for a constant trade-off between training error and margin (c) and cost-factor (j). The similar process was repeated for SubTree kernel and partial tree kernel.

Fig. 5 (a-c). shows the trade-off between training error and margin (c) versus performance (measured by f1-measure) curve for different types of QC data set using different Cost Factor (j) and fixed decay factor (λ=0.01).

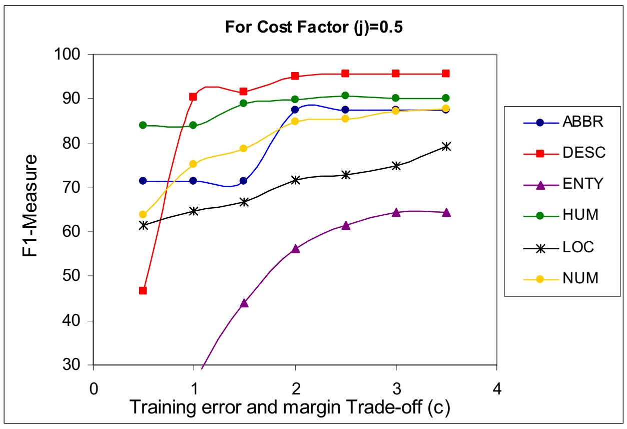

Figure. 5 (a). Training error and Margin Vs F1 Measure for j = 0.5 and SST+BOW

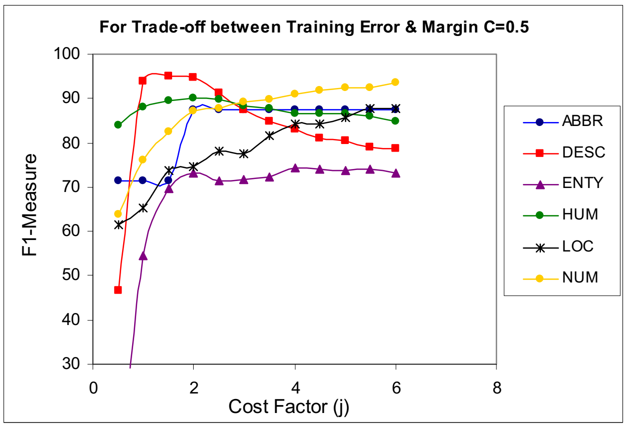

Figure. 6. (a) Cost factor Vs F1 measure for C=0.5 and SST+BOW.

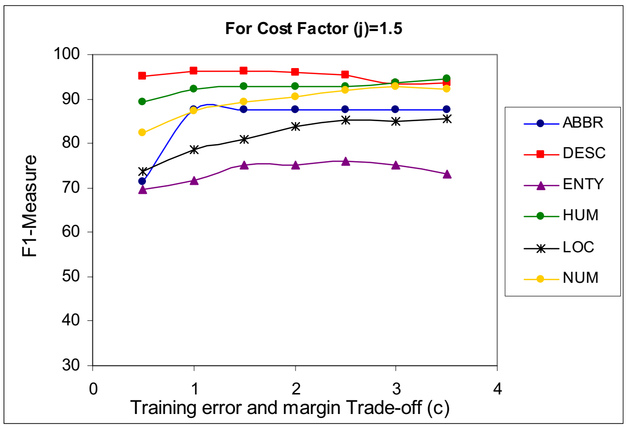

Figure. 5 (b). Training error and Margin Vs F1 Measure for j = 1.5 and SST+BOW

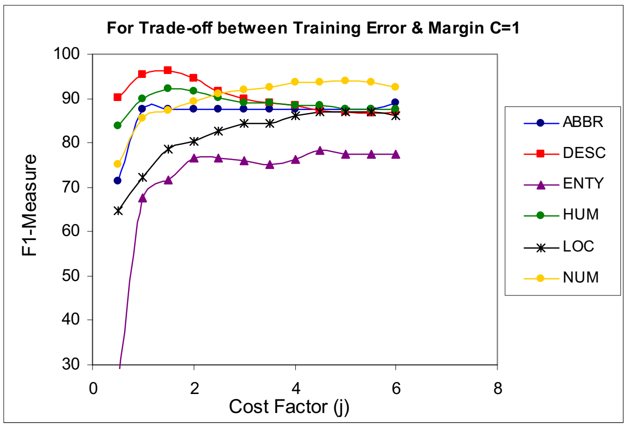

Figure. 6 (b). Cost factor Vs F1 measure for C=1 and SST+BOW.

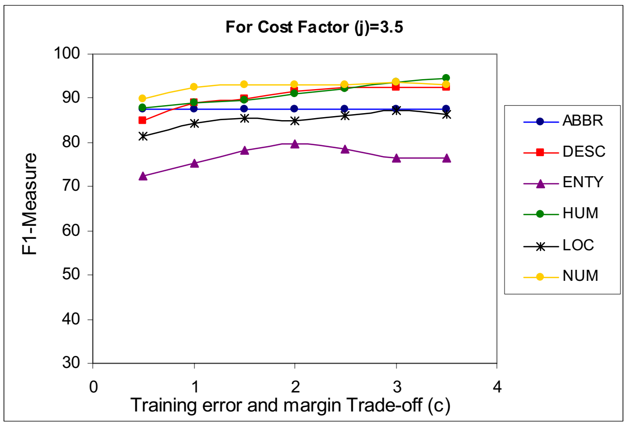

Figure. 5 (c). Training error and Margin Vs F1 Measure for j = 3.5 and SST+BOW

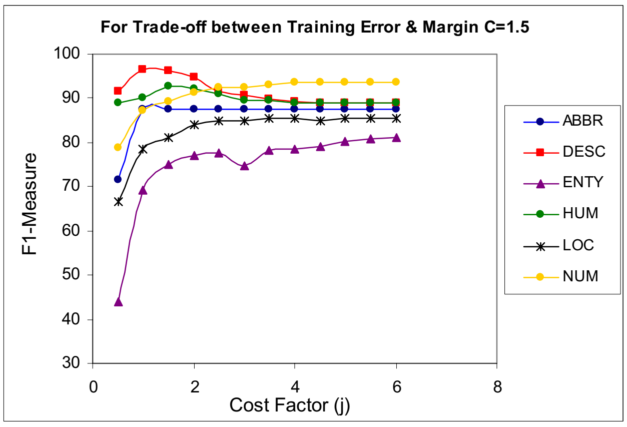

Figure. 6 (c). Cost factor Vs F1 measure for C=1.5 and SST+BOW.

Here it is notable for same set of parameters different data types have different performance level. For different cost factor highest f1-measure can be achieved by setting the trade-off between training error and margin (c) around 2, Afterwards its performance becomes almost constant.

Fig. 6. (a-c) shows the cost factor (j) versus performance (measured by f1-measure) curve for different types of question classification data set using different of Trade of between training error and margin (c) and fixed decay factor (λ=0.01). For a constant trade-off between training error and margin different types of questions have highest f1-measure.

For different value of c highest f1-measure can be achieved by setting the cost factor (j) less than 2. Even for some cases if the value of cost factor is increased then the f1-measue is decreased a certain amount and after a stage it becomes almost constant.

Fig. 7. shows the decay factor (λ) versus f1-measure curve for a constant values of trade-off between training error and margin (c=1.5) and cost factor (j=1.5). To get a very high f1 measure decay factor should have very small. Starting after 0, up to 0.5 its performance is almost steady but if we increase the decay factor more the performance decreased drastically.

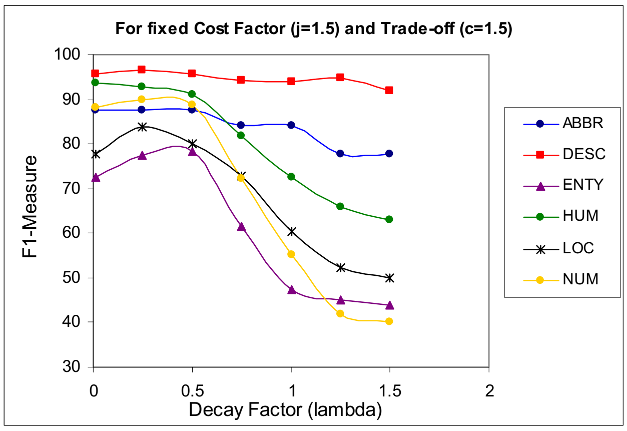

Figure. 7. Impact of Decay Factor and SST+BOW.

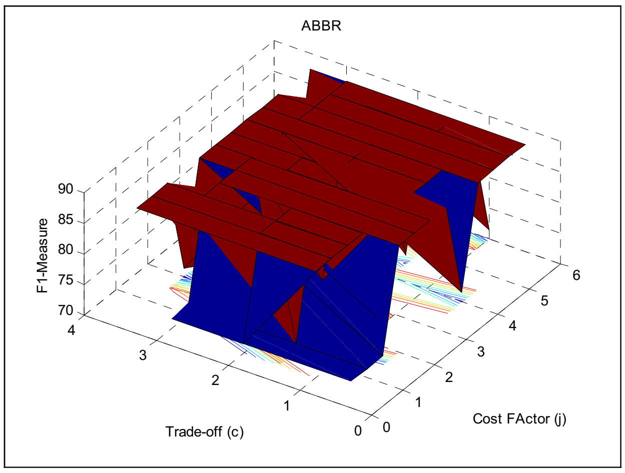

Figure. 8(a). Hyper plane for ABBR data type and SST+BOW.

Fig. 8(a-f). show the hyper plane by which maximum f1-measure can be achieved. For a constant decay factor (λ) a set of values of trade-off between training error and margin (c) and cost factor (j) can provide the highest f1-measure. Among the six type of data set 'LOC' type of question have to classified by some fixed combination of trade-off between training error and margin (c) and cost factor (j) because there is no hyper plane but some peaks.

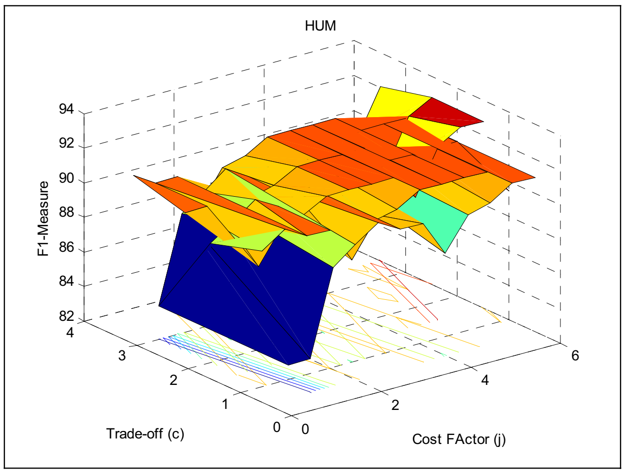

Figure. 8(b). Hyper plane for HUM data type and SST+BOW.

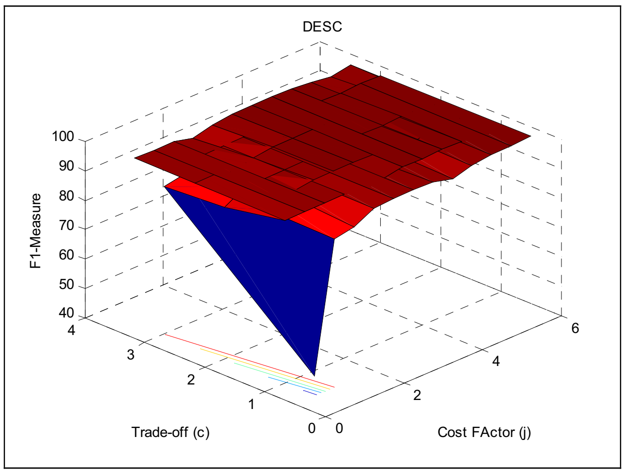

Figure. 8(c). Hyper plane for DESC data type and SST+BOW.

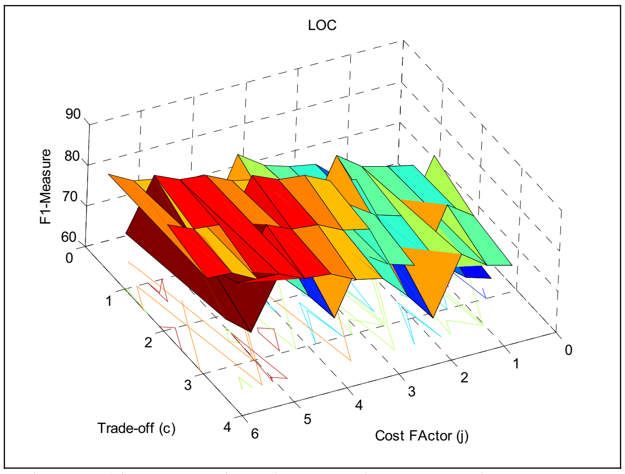

Figure. 8(d). Hyper plane for LOC data type and SST+BOW.

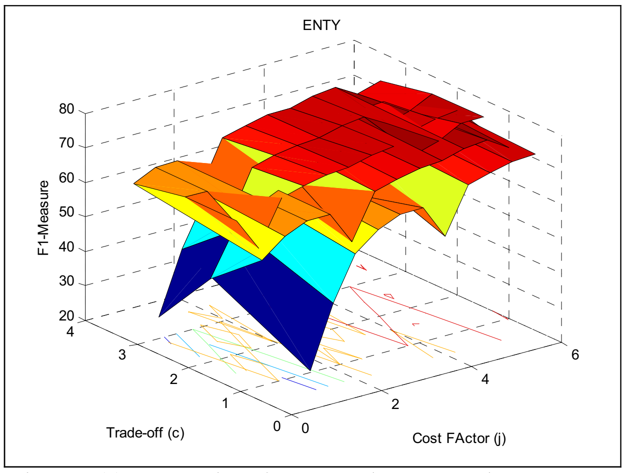

Figure. 8(e). Hyper plane for ENTY data type and SST+BOW.

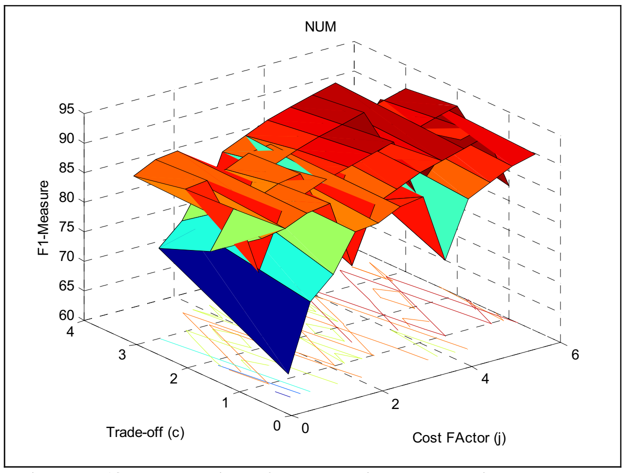

Figure. 8(f). Hyper plane for NUM data type and SST+BOW.

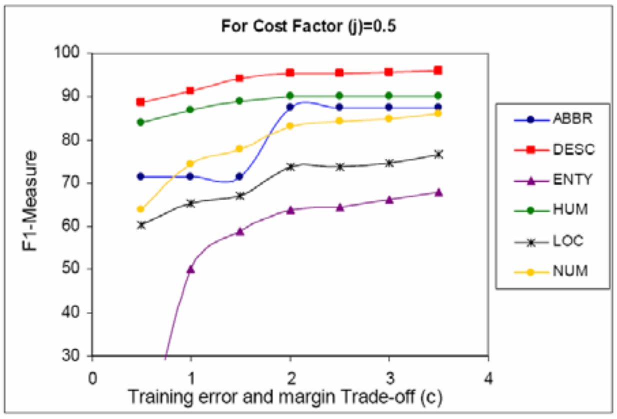

Figure. 9. Training error and Margin Vs F1 Measure
for j = 0.5 and PT

Fig. 9. shows the training error and margin versus F1-measure curve for a constant cost factor (j = 0.5) using partial tree. Like SST+BOW the performance is very low when the training error and margin is very low. After a certain level (here it is 2) of training error and margin the performance tends to become almost constant.

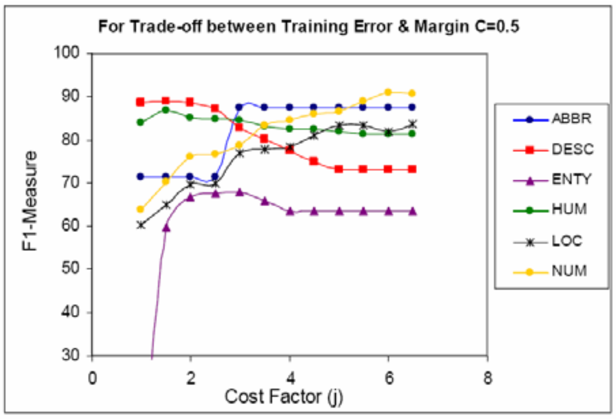

Figure. 10. Cost factor Vs F1 measure for C=0.5 and PT.

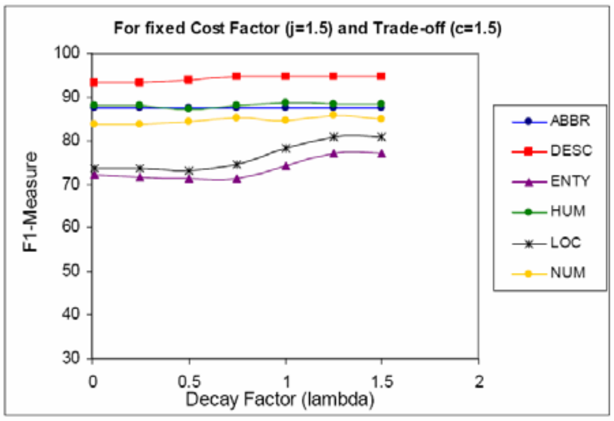

Figure. 11. Impact of Decay Factor for PT.

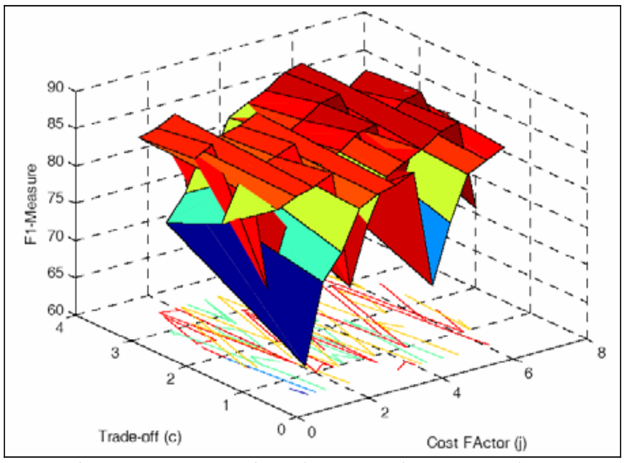

Figure. 12. Hyper plane for NUM data type and PT.

Fig. 10. shows the Cost factor versus F1 measure curve for a constant training error and margin (C=0.5). While Fig. 11. shows the impact of decay Factor. It is clear that at the lower range it doesn't have much impact on performance like SST+BOW. Fig. 12 and Fig.13. respectively shows the hyper plane for NUM and ENTY data type.

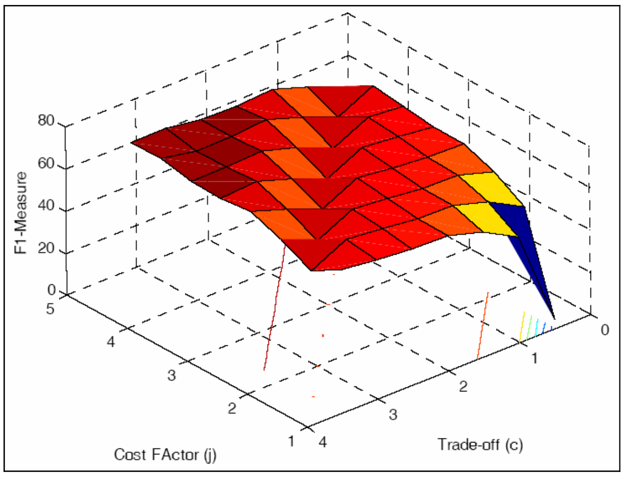

Fig. 13. Hyper plane for ENTY data type and PT.

TABLE II: MAXIMUM F1- MEASURE OBTAINED USING SUBSET TREE KERNEL WITH BAG OF WORDS (SST+BOW) FOR Λ=0.01.

Coarse	F1	P(%)	R(%)	A(%)	j	c
ABBR	88.89	99.6	88.89	88.89	6	1
DESC	96.77	98.2	95.74	97.83	1	2.5
ENTY	82.87	93.8	86.21	79.79	6	2
HUM	95.24	98.8	98.36	92.31	4	3.5
LOC	87.8	96	86.75	88.89	5.5	0.5
NUM	94.01	97.4	98.08	90.27	5	1

TABLE III: MAXIMUM F1- MEASURE OBTAINED USING SUBSET TREE KERNEL FOR Λ=0.01.

Coarse	F1	P(%)	R(%)	A(%)	j	c
ABBR	88.89	99.6	88.89	88.89	3.5	2.5
DESC	97.12	98.4	96.43	97.83	1	3.5
ENTY	79.78	92.8	84.52	75.53	6.5	3
HUM	93.75	98.4	95.24	92.31	2	2.5
LOC	88.48	96.2	86.9	90.12	5.5	1
NUM	94.01	97.4	98.08	90.27	5	1.5

TABLE IV: MAXIMUM F1- MEASURE OBTAINED USING PARTIAL TREE KERNEL FOR Λ=0.01.

Coarse	F1	P(%)	R(%)	A(%)	j	c
ABBR	87.5	99.6	100	77.78	3	0.5
DESC	96.06	97.8	95.04	97.1	1	3.5
ENTY	85.56	94.6	86.02	85.11	4	3.5
HUM	81.33	94.4	71.76	93.85	5.5	0.5
LOC	88.48	96.2	86.9	90.12	5.5	1.5
NUM	94.44	97.6	99.03	90.27	4.5	3

Table II shows the maximum F1- measure (F1) obtained using SubSet tree Kernel-Bag of words (SST+BOW) for λ=0.01 and the corresponding precision (P), recall (R), accuracy (A) for specific cost factor (j) and trade-off between training error and margin (c). Table III and Table IV shows the performance for SubTree kernel and Partial Tree kernel respectively.

VI. CONCLUSION

Question classification is one importance role in the Question Answering frame to reduce the gap between question and answer. It can conduct answer choosing and selection. Our question classification method based on the use of linguistic knowledge and machine learning approaches and it exploit different classification features and combination method, also. Though among all the experiments one or two data set did not provide distinguishable hyper plane in every cases thereafter the outcome of experiments done using the tool SVM_light on Li and Roth question classification data sets demonstrate some optimal set of values which can maximize the performance. In the future we aim to investigate the impact of different parameters on constituent trees using different types of kernel as well as other different classifiers.

REFERENCES

[1]. Hovy, E., Hermjakob, U., Lin, C.-Y. and Ravichandran, D. "Using Knowledge to Facilitate Factoid Answer Pinpointing", *Proceedings of the 19th International Conference on Computational Linguistics (COLING),* Taipei, Taiwan, 2002.

[2]. Ittycheriah, A., Franz, M., Zhu, W.-J. and Ratnaparkhi, A. "IBM's Statistical Question Answering System", *Proceedings of the TREC-9 Conference*, Gaithersburg, MD: NIST, 2000, p. 229.

[3]. Zhang, D. and Lee, W. S. 2003. "Question Classification using Support Vector Machines", *Proceedings of ACMSIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Toronto, Canada, 2003.

[4]. LI Xin, HUANG Xuan-Jing, WU Li-de, "Question Classification by Ensemble Learning", *IJCSNS International Journal of Computer Science and Network Security*, VOL.6 No3, March 2006.

[5]. Alessandro Moschitti, Silvia Quarteroni, Roberto Basili and Suresh Manandhar, "Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classification", *Proceedings of the 45th Conference of the Association for Computational Linguistics* (ACL), Prague, June 2007.

[6]. Roberto Basili, Alessandro Moschitti, *"Automatic Text Categorization From information Retrival to Support Vector Learning"*, ARACNE editrice, November 2005.

[7]. Stephan Bloehdorn and Alessandro Moschitti, "Exploiting Structure and Semantics for Expressive Text Kernels", *Proceeding of the Conference on Information Knowledge and Management*, Lisbon, Portugal, 2007.

[8]. Li, X., Roth, D, "Learning question classifiers", *Proceedings of the 19th International Conference on Computational Linguistics,* Taipei, Taiwan, 2002.

[9]. M.-Y. Day, C.-H. Lu, C.-S. Ong, S.-H. Wu, and W.-L. Hsu, "Integrating Genetic Algorithms with Conditional Random Fields to Enhance Question Informer Prediction", *Proceedings of the IEEE International Conference on Information Reuse and Integration*, Waikoloa, Hawaii, USA, 2006, pp. 414-419.

[10]. V. Vapnik, "*Statistical Learning Theory*", Wiley, New York, USA 1998.

[11]. K. Morik, P. Brockhausen, and T. Joachims, "Combining statistical learning with a knowledge-based approach – A case study in intensive care monitoring", *International Conference on Machine Learning (ICML)*, Bled, Slovenia, 1999.

[12]. M. Collins, N. Duffy, "New Ranking algorithm for parsing and Tagging: Kernels over Distance structure, and the Voted Perception", *Association for Computational Linguistics (ACL)*, Philadelphia, USA, 2002.

[13]. Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, Christopher Watkins, "Text Classification using string kernels", *in NIPS*, pp 563-569, 2000.

Muhammad Arifur Rahman has completed his Masters degree in Human Language Technology and Interfaces (HLTI) under the Department of Information Engineering and Computer Science at University of Trento, Italy. He received his B.Sc. (Honors) degree from the Department of Computer Science and Engineering, Jahangirnagar University, Bangladesh at 2001. His major areas of interest include Natural Language Processing, Data Communication, and Digital Signal Processing.

He has authored a book titled "Fundamentals of Communication" and more than 10 international journal and conference papers. At present he is serving as a Lecturer in the Department of Physics, Jahangirnagar University, Dhaka, Bangladesh.