N-gram based text classification for Persian newspaper corpus

Article · January 2011

CITATIONS
9

READS
630

3 authors, including:

Mojgan Farhoodi
Iran Telecommunication Research Center
18 PUBLICATIONS **108** CITATIONS
SEE PROFILE

Alireza Yari
Iran Telecommunication Research Center
31 PUBLICATIONS **141** CITATIONS
SEE PROFILE

N-Gram Based Text Classification for Persian Newspaper Corpus

Mojgan Farhoodi, Alireza Yari and Ali Sayah

Iran Telecommunication Research Center
{farhoodi, yari}@itrc.ac.ir

***Abstract*—** Statistical n-gram language modeling is applied in many domains like speech recognition, language identification, machine translation, character recognition and topic classification. Most language modeling approaches work on n-grams of words. In this paper, we employ language models classifier based on word level n-grams for Persian text classification. The presented approach computes the occurrence probability on word sequence in training data. Then by extracting the word sequence in test data, it can predict the highest probability for related class to given news text. We show that statistical language modeling can significantly cause high classification performance. The experimental results on Hamshahri corpus show satisfactory results and n-grams of length 3 are the most useful for Persian text classification.

Keywords- Persian text classification, N-gram, language modeling , Hamshahri courpus, Smoothing methods.

I. INTRODUCTION

Text classification addresses the problem of assigning a given passage of text (or a document) to one or more predefined classes. This is an important area of information retrieval research that has been heavily investigated, although most of the research activity has concentrated on English text [1,2,3,4]. Text classification in Asian languages such as Persian, however, is also an important (and relatively more recent) area of research that introduces a number of additional difficulties. One difficulty with Persian text classification is that, unlike English, Persian texts do not have explicit whitespace between words. This means that some form of word segmentation is normally required before further processing. However, word segmentation itself is a difficult problem in this language. A second difficulty is that in Persian language, finding the root of words is a complex task. Also we don't have a good benchmark data set for this language [5].

Many standard machine learning techniques have been applied to text categorization problems, such as naïve Bayes classifiers [6,7], support vector machines (SVM) [8,9,10], neural networks[11], and k-nearest neighbor (KNN) classifiers [12,13]. A common aspect of these approaches is that they treat text categorization as a standard classification problem, and thereby reduce the learning process to two simple steps: feature engineering, and classification learning over the feature space. Of these two steps, feature engineering is critical to achieving good performance in text categorization problems. Once good features are identified, almost any reasonable technique for learning a classifier seems to perform reasonably well. Unfortunately, the standard classification learning methodology has several drawbacks. First, there are an enormous number of possible features to consider in text categorization, and standard feature selection approaches do not always cope well in such circumstances. For example, given a sufficiently large number of features, the cumulative effect of uncommon features can still have an important effect on classification accuracy, even though infrequent features contribute less information than common features individually. Therefore, throwing away uncommon features is usually not an appropriate strategy in this domain [14]. Another problem is that feature selection normally uses indirect tests, such as X^2 or mutual information, which involve setting arbitrary thresholds and conducting a heuristic greedy search to find a good subset of features. Moreover, by treating text categorization as a classical classification problem, standard approaches can ignore the fact that texts are written in natural language, which means that they have much implicit regularity that can be well modeled by specific tools from natural language processing.

In this paper, we propose a simple text categorization approach based on statistical n-gram language modeling to overcome the above shortcomings. An advantage we exploit is that the language modeling approach does not discard low frequency features during classification, as is commonly done in traditional classification learning approaches. Also, the language modeling approach uses n-gram models to capture more contextual information than standard bag-of-words approaches, and employs better smoothing techniques than standard classification learning.

In oriental languages such as Persian, there are two obvious and distinct ways to perform text classification: character based and word based. In the word based approach, one first must segment the character sequence into individual words, and then apply standard classification techniques from word based textual languages, such as English. In the character based approach, by contrast, one instead works directly with character level n-gram features and side steps the segmentation problem. However, although Persian text classification can be conducted entirely at the character n-gram level, it is still not clear that this is a better approach than taking word segmentation information into account as well. Moreover, if word segmentation information is useful, one is still left with the question of determining the relationship between classification performance and segmentation accuracy. Answering these questions is of theoretical and practical importance in designing Persian text classification systems.

[15] presents an experiment of classification of Persian documents by using the Learning Vector Quantization network. In this method, each class is presented by an exemplar vector called codebook. The codebook vectors are placed in the feature space in a way that decision boundaries are approximated by the nearest neighbor rule. [16] presents the results of automated classifying Farsi text documents using tri-gram, quad-gram, and word frequency statistics methods. [17, 18] use the Bayesian approach for Persian documents and they improve it by using the word collocation.

In this paper the behavior of the N-Gram Frequency Statistics technique for classifying Persian text documents is studied. In this regard, we used Hamshahri dataset. All documents, whether training documents or documents to be classified went through a preprocessing phase removing punctuation marks, stop words, diacritics, and non letters. For the training documents, the N-gram (N=1-4) frequency profile was generated for each document and saved in text files. Then for each document to be classified, the N-gram frequency profile was generated and compared against the N-gram frequency profiles of all the training classes. Results show that 3-gram text classification gives better classification results compared to the other n-grams.

The rest of the paper is organized as follows. First, in section 2, we describe the text classification method. In section 3, language model text classifier has been explained. We express our method for Persian text classification in Section 4. Section 5 shows the results of experiment Persian classifier on Hamshahri corpus. Finally in section 6, we have concluded the paper and explained about the future work.

II. TEXT CLASSIFICATION METHOD

Text classification is the problem of assigning a document d to one of a set of |C| pre-defined categories C= $\{c_1, c_2, \dots, c_{|C|}\}$. Normally a supervised learning framework is used to train a text classifier, where a learning algorithm is provided a set of N labeled training examples$\{(d_i, c_i): i = 1, \dots, N\}$) from which it must produce a classification function $F: D \rightarrow C$ that maps documents to categories. Here d_i denotes the ith training document and c_i is the corresponding category label of d_i. We use the random variables D and C to denote the document and category values respectively. A probabilistic text classifier is formulated as the following decision problem: given a document d, determine the class label c^* that yields the highest posterior probability $P(C = c|D = d)$ (written $P(c|d)$ for simplicity):

$$c^* = \arg\max_{c \in C}\{P(c|d)\} \quad (1)$$

III. LANGUAGE MODEL TEXT CLASSIFIER

We now present a language modeling based text classifier. The goal of language modeling is to predict the probability of natural word sequences; or more simply, to put high probability on word sequences that actually occurs (and low probability on word sequences that never occur). Given a word sequence $w_1 w_2 \dots w_T$ to be used as a test corpus, the quality of a language model can be measured by the empirical perplexity (or entropy) on this corpus [14]:

$$Perplexity = \sqrt[T]{\prod_{i=1}^{T} \frac{1}{P(w_i|w_1 \dots w_{i-1})}} \quad (2)$$

$$Entropy = log_2 Perplexity \quad (3)$$

The goal of language modeling is to obtain a small perplexity.

A. N-Gram Language Modeling

The simplest and most successful basis for language modeling is the n-gram model. Note that by the chain rule of probability we can write the probability of any sequence as [14]:

$$P(w_1 w_2 \dots w_T) = \prod_{i=1}^{T} P(w_i|w_1 \dots w_i w_{i-1}) \quad (4)$$

An n-gram model approximates this probability by assuming that the only words relevant to predicting $P(w_i|w_1 \dots w_i w_{i-1})$ are the previous n-1 words; that is, it assumes the Markov n-gram independence assumption:

$$P(w_i|w_1 \dots w_{i-1}) = P(w_i|w_{i-n+1} \dots w_{i-1}) \quad (5)$$

A straightforward maximum likelihood estimate of n-gram probabilities from a corpus is given by the observed frequency:

$$P(w_i|w_{i-n+1} \dots w_{i-1}) = \frac{\#(w_{i-n+1} \dots w_i)}{\#(w_{i-n+1} \dots w_{i-1})} \quad (6)$$

where #(.) is the number of occurrences of a specified gram in the training corpus. Unfortunately, using grams of length up to n entails estimating the probability of W^n events, where W is the size of the word vocabulary. This quickly overwhelms modern computational and data resources for even modest choices of n. Also, because of the heavy tailed nature of

language (i.e. Zipf's law) one is likely to encounter novel n-grams that were never witnessed during training. In [1], Cavnar and Trenkle summarize Zipf's Law as "*The n^{th} most common word in a human language text occurs with a frequency inversely proportional to n*". That is, $f \propto \frac{1}{r}$, where f is the frequency of the word and r is the rank of the word in the list ordered by the frequency [19]. Therefore, some mechanism for assigning non-zero probability to novel n-grams is a central and unavoidable issue. Some standard approaches to smoothing probability estimates to cope with sparse data problems (and to cope with potentially missing n-grams) are add-one, absolute discounting and back-off estimator that explained in part C.

B. Language Models as Text Classifiers

Text classifiers attempt to identify attributes which distinguish documents in different categories. Such attributes may include vocabulary terms, word average length, local *n*-grams, or global syntactic and semantic properties. Language models also attempt capture such regularities, and hence provide another natural avenue to constructing text classifiers. An n-gram language model can be applied to text classification in a similar manner to a naïve Bayes model. In this case [14]:

$$c^* = \arg\max_{c \in C}\{P(c|d)\} = \arg\max_{c \in}\{P(d|c)P(c)\} \quad (7)$$

$$= \arg\max_{c \in C}\{P(d|c)\} \quad (8)$$

$$= \arg\max\{\prod_{i=1}^{T} P_c\,(w_i|w_{i-n+1} \dots w_{i-1})\} \quad (9)$$

where the step from equation (7) to equation (8) assumes a uniform prior over categories, and the step from equation (8) to equation (9) uses the Markov *n*-gram independence assumption. Likelihood is related to perplexity and entropy by equation (2) and equation (3). The principle for using an *n*-gram language model as a text classifier is to determine the category that makes a given document most likely to have been generated by the category model (equation (9)). Thus, we train a separate language model for each category, and classify a new document by evaluating its likelihood under each category, choosing the category according to equation (9). A pure multinomial naïve Bayes is a special case of n-gram based text classifier where n=1 and add-one smoothing is used. However, *n*-gram modeling based approach considers more context information and deals with unobserved attributes with back-off approach, coupled with better smoothing techniques than add-one smoothing. These advantages will be shown in experiments.

C. Smoothing Methods

The term of smoothing describes techniques for adjusting the maximum likelihood estimate of probabilities to produce more accurate probabilities. The name comes from the fact that these techniques tend to make distribution more uniform, by adjusting low probabilities such as zero probabilities upward, and high probabilities downward. Not only do smoothing methods generally prevent zero probabilities, but they also attempt to improve the accuracy of the model as a whole [14].

Some of important methods for smoothing are described below [11, 20]:

C.1. Add-one

One of the simplest types of smoothing used in practice is add-one smoothing. To avoid zero probabilities, we pretend that each *n*-gram occurs slightly more often than it actually does: this method adds 1 to every count. The model is given by

$$P_{add-one}(x_i|x_{i-n+1}^{i-1}) = \frac{C(x_{i-n+1}^{i}) + 1}{C(x_{i-n+1}^{i-1}) + V} \quad (10)$$

where x_i^j denotes the sequence $x_i \dots x_j$, V is the size of the vocabulary (number of different words in the language) and $C(x)$ denotes the number of occurrences of a word *x*.

C.2. Absolute Discounting

The idea of the absolute discounting method is to lower the probability of seen words by subtracting a constant from their counts. The model is given by

$$P_{abs}(x_i|x_{i-n+1}^{i-1}) = \frac{C(x_{i-n+1}^{i}) - D}{C(x_{i-n+1}^{i-1})} + \delta_{x_{i-n+1}^{i-1}} P(x_i|x_{i-n+1}^{i-1}) \quad (11)$$

where $\delta_{x_{i-n+1}^{i-1}}$ is a scaling factor that makes the conditional distribution sum to one.

C.3. Back-off

The idea of back-off smoothing is going back to "smaller" *n*-grams. For example, in 3-gram language modeling it doesn't only use trigram probability, but also use bigrams and unigrams probabilities. In this method, if no trigram found, it uses bigram and if no bigram found, it uses unigram. The model is given by:

$$P_{bo}(x_i|x_{i-2}, x_{i-1}): \begin{cases} (1 - d(x_{i-2}, x_{i-1}))P(x_{i-2}, x_{i-1}) \\ \quad \text{if } \text{count}(x_{i-2}, x_{i-1}) > 0 \\ \alpha(x_{i-2}, x_{i-1})P_{bo}(x_i|x_{i-1}) \\ \quad otherwise \end{cases} \quad (12)$$

IV. Persian Text Classification

In order to applying the above method in Persian news classification, it is necessary to have some lingual preprocessing such as: Text segmentation, word segmentation, normalizing, eliminating the stop words and word stemming.

Text segmentation, one of the primary activities in text preprocessing, is the process of recognizing boundaries of text constituents, such as paragraphs, sentences, phrases and words. Word segmentation also known as tokenization focuses on

recognizing word boundary delimiters, punctuation marks, written forms of alphabet and affixes. The developed tokenizer determines words boundaries as explained in [22].

To achieve the goal of Persian text classification, after removing the xml tags, whole words of each document are extracted by using a Persian tokenizer. Then in word segmentation process, all stop words are eliminated from the extracted tokens. The main reason for elimination stop words is that the stop words frequently occur in all corpora, and they usually don't have any added value in the process of text classification.

There are some letters such as 'ی' (i) and 'ک' (k) for which we have two Unicode (one for Persian and one for Arabic). In Persian text both are used. In normalizing step, we have to unify their occurrences. In this paper, encoding the all text files converted to UTF-8.

In addition, there are some imported sounds such as "Tanwin" and "Hamza" from Arabic which we use in some imported words in Persian. For example 'پاییز' and 'پائیز' are different forms of writing the word 'fall' in Persian. The required lingual preprocessing here, is the unification of these kinds of words.

In the last step of preprocessing, suffixes and prefixes of each word are removed and stems of them are extracted by Persian stemmer. Thereafter, the word sequence of each document is extracted in a vector which is containing whole words in the document sequentially.

After language preprocessing, now in the Persian text classifier, the language modeling is applied for deciding about document classes. As explained above in section 3, the probabilistic distribution is estimated for each class by using the training set. Then, to classify each sample text, the classifier first creates the document vector and then extracts word sequences, finally compares the language model of each sample to the language model of each class and return the class with the highest probability as a result.

V. Experiment and Results

In this section, first we explain about Hamshahri dataset and then describe our experimental results for Persian N-gram based text classification.

A. Hamshahri Dataset

Hamshahri dataset [23] produced according to CLEF standard in Tehran university research group. Hamshahri is one of the most popular daily newspapers in Iran that has been publishing for more than 20 years. Hamshahri corpus is a Persian test collection that consists of news texts from this newspaper since 1996 to 2007. This corpus contains more than 300,000 news articles about variety of subjects [3]. Hamshahri articles vary between 1 KB and 140 KB in size. The categories are however overlapping and non-exhaustive, and there are relationships among the categories. Therefore, in order to avoid ambiguities, classes are merged to 9 final categories prior to training. Mentioned text documents have been stored in XML format and UTF-8 standard.

B. Emperical Evaluation and Analysis

We now present our experimental results about the proposed Persian text classifier. The Persian dataset we used is a part of Hamshahri dataset. Our dataset consists of 9000 Persian documents of different lengths that belong to 9 categories. The categories are: Literature and Art, Social, Science and Culture, Miscellaneous, Politics, Sport, Natural Environment, Economy, Tourism. For each category we randomly select about 1000 sample documents. By using an XML parser, we saved each document of news in a separate text file.

B.1. Influence of Linguistic Preprocessing

In order to investigating the effect of lingual preprocessing in classification performance, we get the accuracy without and with considering the preprocessing. The intention of linguistic preprocessing in this paper is normalizing, eliminating the stop words, tokenizing and stemming.

Table 1 and Table 2 show these results respectively:

Table 1- Classification accuracy without considering linguistic preprocessing

Number of Dataset Sample	1-gram	2-gram	3-gram	4-gram
1000	0.47	0.81	0.84	0.846
3000	0.49	0.84	0.85	0.85
5000	0.51	0.87	0.87	0.88
9000	**0.61**	**0.90**	**0.92**	**0.925**

Table 2- classification accuracy with considering linguistic preprocessing

Number of Dataset Sample	1-gram	2-gram	3-gram	4-gram
1000	0.42	0.87	0.91	0.92
3000	0.43	0.87	0.92	0.93
5000	0.54	0.89	0.96	0.96
9000	**0.67**	**0.96**	**0.98**	**0.984**

B.2. Influence of the n-gram Order

The order n is a key factor in n-gram language modeling. An order n that is too small will not capture sufficient information to accurately model word dependencies. On the other hand, a context n that is too large will create sparse data problems in training. In our experiments, we did not observe significant improvement when using higher order n-gram models (n>3). In fact, we observed an immediate decrease in performance for the word level model, due to the early onset of sparse data problems. For solving this problem, we use of smoothing methods, hence, the accuracy of results which presented in Table1 and Table 2 calculated by applying the back-off smoothing. Also if more training data were available, the higher order models may begin to show an advantage. For example, in the larger dataset (average 1000 documents per

class for training) we observe an obvious increase in classification performance with higher order models (Table2). However, it is valuable to mention when n becomes too large, overfitting will begin to occur.

B.3. Influence of Smoothing Techniques

Smoothing plays a key role in language modeling. In the case we have examined, add-one smoothing is obviously the worst smoothing technique, since it systematically overfits much earlier than the more sophisticated smoothing techniques. Back-off smoothing makes the better accuracy in our dataset.

Fig 1 shows these results.

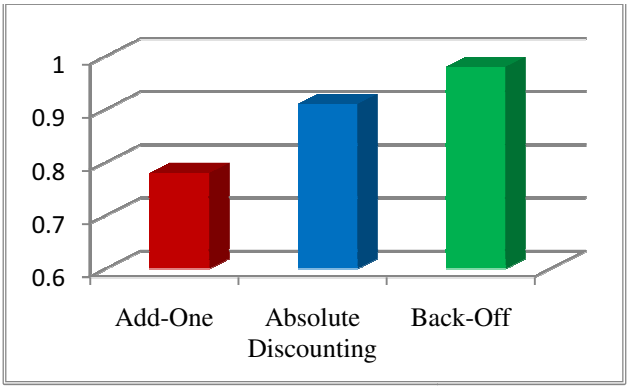

Figure 1: 3-gram classifier accuracy with different smoothing methods

VI. CONCLUSION AND FUTURE WORK

We have presented a simple language model based approach for Persian text classification using word level *n*-grams. It performs quite well in assigning topics to the text of Persian news domain. We show the classification accuracy in the word based approach is better than the character based approach represented in [16]. We have also shown how the *n*-gram based modeling will be performed in accordance to number of grams with or without linguistic preprocessing. According to the results of our experiments, the 3-gram language modeling has the best accuracy in the Persian text classification. It becomes even better when we apply back-off smoothing method for unseen events.

Future research will try to extend the approach in a way that allows the automatic assigning of multiple topics to a document. A possibility would be to learn thresholds values based on the corresponding probabilities of the topics ranking. We will also work on combining the machine learning algorithm with n-gram based algorithm to improve the text classification.

REFERENCES

[1] W.B. Cavnar and J.M. Trenkle, "N-Gram-Based Text Categorization", In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, 1994.

[2] P. Nather, "N-gram Based Text Categorization", Diploma thesis, 2005

[3] A. AleAhmad, P. Hakimian, F. Oroumchian, N-gram and local context analysis for persian text retrieval, International Symposium on Signal Processing and its Applications, Sharjah,United Arab Emirates (UAE), 2007.

[4] P. Taheri Makhsoos, M. R. Kangavari, H. R. Shayegh, "Improving Feature Vector by Words' Position and Sequence for Text Classification", International Conference on IT to Celebrate S.Charmonman's 72nd Birthday, 2009.

[5] S. Kiani, M. Shamsfard, "Word and Phrase Boundary detecttion in Persian Texts", 14th CSI Computer Conference, Iran, 2008.

[6] F. Colace, M. De Santo, "A Bayesian Approach for Text Classification", IEEE, pp. 1323-1326, 2006

[7] M. El. Kourdi, A. Bensaid, T. Rachidi, "Automatic Arabic Document Categorization Based on the Naïve Bayes Algorithm", in Proc. of COLING 20th Workshop on Computational Approaches to Arabic Script-based Languages, 2004.

[8] A. M. Mesleh, Gh. Kanaan, "Support Vector Machine Text Classification System: Using Ant Colony Optimization Based Feature Subset Selection", IEEE, pp 143-148, 2008

[9] E. Leopold, J. Kindermann, "Text Categorization with Supprt Vector Machines. How to Represent Texts in Input Space?", 2002

[10] Z. Wang, X. Sun, D. Zhang, X. Li, "An Optimal SVM-Based Text Classification Algorithm", of the Fifth International Conference on Machine Learning and Cybernetics, Dalian, 2006

[11] S. Ramasundaram, S.P. Victor, "Text Categorization by Backpropagation Network", International Journal of Computer Applications, 2010

[12] R. Al-Shalabi, Gh. Kanaan, "Arabic Text Categorization Using KNN algorithm", Proceedings of The 4th International Multiconference on Computer Science and Information Technology, Vol. 4, ., 2006.

[13] Gh. Kanaan, R. Al-Shalabi, A. Al-Akhras, "KNN Arabic Text Categorization Using IG Feature Selection", Proceedings of The 4th International Multiconference on Computer Science and Information Technology, Vol. 4, 2006.

[14] F. Peng, X. Huang, "Machine Learning for Asian Language Text Classification", 2006

[15] M. T. Pilevar, H. Feili, M. Soltani, "Classification of Persian Textual Documents Using Learning Vector Quantization", 4rd IEEE Conference on Knowledge Engineering and Natural Language Processing, NLP-KE, 2009.

[16] B. Bina, M. H. Ahmadi, M. Rahgozar, "Farsi Text Classification Using N-Grams and KNN algorithm A cpmarative Study", in Proc. DMIN, pp.385-390, 2008.

[17] H. Faili, M. Arabsorkhi, Persian Text Classification using supervised approach, CSI Computer Conference (CSICC'2009), Tehran, Iran, (in poster, in Persian).

[18] M. Arabsorkhi, H. Faili, Using Bayesian Model to Persian Text Classification, in the 2nd Workshop on Persian Language and Computer, Tehran University, Tehran, Iran, 2006.

[19] M. Mansur, N. UzZaman, M. Khan, "Analysis of N-Gram Based Text Categorization for Bangla in a Newspaper Corpus", 2006.

[20] Stanley F. Chen, Joshua Goodman, "An Emperical Study of Smoothing Techniques for Language Modeling", 1999

[21] T. Vatenan, J. J. Vayrynen, S. Virpioja, "Language Identification of Short Text Segments with N-gram Models", LREC 2010

[22] M. Shamsfard, S. Kiani, Y. Shahedi, "STeP-1: Standard Text Preparation for Persian Language", CAASL3 Third Workshop on Computational Approaches to Arabic Script-based Languages, Canada, 2009.

[23] http://ece.ut.ac.ir/DBRG/Hamshahrifa/