

# URL-Based Web Page Classification: With n-Gram Language Models

Tarek Amr Abdallah and Beatriz de La Iglesia<sup>(✉)</sup>

School of Computing Sciences, University of East Anglia, Norwich Research Park,  
Norwich NR4 7TJ, UK  
[b.Iglesia@uea.ac.uk](mailto:b.Iglesia@uea.ac.uk)

**Abstract.** There are some situations these days in which it is important to have an efficient and reliable classification of a web-page from the information contained in the Uniform Resource Locator (URL) only, without the need to visit the page itself. For example, a social media website may need to quickly identify status updates linking to malicious websites to block them. The URL is very concise, and may be composed of concatenated words so classification with only this information is a very challenging task. Methods proposed for this task, for example, the all-grams approach which extracts all possible sub-strings as features, provide reasonable accuracy but do not scale well to large datasets.

We have recently proposed a new method for URL-based web page classification. We have introduced an n-gram language model for this task as a method that provides competitive accuracy and scalability to larger datasets. Our method allows for the classification of new URLs with unseen sub-sequences. In this paper we extend our presentation and include additional results to validate the proposed approach. We explain the parameters associated with the n-gram language model and test their impact on the models produced. Our results show that our method is competitive in terms of accuracy with the best known methods but also scales well for larger datasets.

**Keywords:** Language models · Information retrieval · Web classification · Web mining · Machine learning

## 1 Introduction

During 2010 twitter users sent about 90 million updates every day, as reported by Thomas et al. [1]. It is estimated that 25% of those updates contain web-links. Similarly, a huge number of links are carried by the millions of email messages and Facebook updates sent every day. In such context, it is crucial to be able to classify web-pages in real-time using their URLs only, without the need to visit the pages themselves, even if some accuracy is sacrificed for the sake of greater speed of classification. Also, search engines depend mainly on textual data to retrieve on-line resources. However, they are often faced with multimedia content such as videos and images with scarce descriptive tags or

surrounding text. Thus, in this context, URL-based classification can be used to decide the categories of such content enhancing the retrieval performance.

Additionally, the classification approach presented here is not limited to URL-based classification tasks only. It can also be adapted for similar problems where there is a need to classify very concise documents with no obvious boundaries between words, e.g. social networks folksonomies.

Unlike documents, URLs are very concise as they are composed of very few words. Usually, words are also concatenated without intermediate punctuations or spaces; for example: [carsales.com](http://carsales.com) and [vouchercodes.co.uk](http://vouchercodes.co.uk). They also contain various abbreviations and domain-specific terms. Therefore, classification requires specific approaches that can deal with the special characteristics of the data under consideration.

## 2 Related Work

Previous researchers have focused on how to extract features from URLs. Early approaches segmented URLs based on punctuation marks using the resulting terms as the classifier’s feature-set [2]. Later on, researchers used either statistical or brute-force approaches to further segment URLs beyond the punctuation marks. The non-brute-force approaches used *information content* [2], *dictionary based tokenizes* [3] and *symmetric/non-symmetric sliding windows* [4]. The brute-force approach, on the other hand, tends to extract all possible sub-strings, *all-grams*, to use them as the classifier’s feature-set [5–8]. To our knowledge, this is the most successful so far, however, it is obvious that it does not scale very well. In our experiments reported here, the resulting datasets from applying the all-grams approach can be very large, going beyond our computational resources and therefore it becomes difficult to store, classify or even to select subsets of features. For example, in a dataset of 43,223 URLs, when extracting all-grams between 4 and 8 characters-long, we ended up with 1,681,223 n-grams as our feature-set.

The aforementioned classification algorithms are sometimes called batch algorithms, as opposed to online algorithms. In recent research, online learners have been used in URL-based classifications [9, 10]. Nevertheless, they incorporate meta-features, such as those obtained from WHOIS and geographic information, in addition to the URLs’ lexical features. We prefer to limit ourselves here to features found in the URLs only.

Our proposed approach tries to classify URLs without the need to segment them. We borrow the concept of language models from the information retrieval and automatic speech recognition field. We apply a similar approach to that used by Peng et al. to classify Japanese and Chinese documents [11]. They used an *n-Gram Language Model (LM)* in order to classify textual data without the need for segmenting into separate terms. These two East Asian languages are similar to URLs in the sense that spaces between words are absent so we hypothesise that a similar approach can work for the URL classification problem. We have adapted the model used by Peng et al to be used with URLs, given their format and

punctuations. Furthermore, we made use of the *Linked Dependence Assumption* to relax the model's independence assumption and to improve its performance. We further expand on this in Sect. 3.2.

In the next section we are going to explain the n-gram Language Model and its use for document classification. In Sect. 4 we give more details on the dataset used, and the experiments done. Then, we present our results in Sect. 5. Finally, we conclude our findings and offer suggestions for future researchers in the last section.

### 3 The n-Gram Language Model

Let us assume we have a set of documents  $D = \{d_1, d_2, \dots, d_m\}$ , and a set of classes  $C = \{c_1, c_2, \dots, c_k\}$ , where each document is classified as member of one of these classes. For any document,  $d_i$ , the probability that it belongs to class  $c_j$ , can be represented as  $Pr(c_j/d_i)$  and using Bayes rules [11, 12], this probability is calculated by:

$$Pr(c_j/d_i) = \frac{Pr(d_i/c_j) * Pr(c_j)}{Pr(d_i)} \quad (1)$$

The term  $Pr(d_i)$  is constant for all documents. The term  $Pr(c_j)$  can represent the distribution of class  $j$  in the training set. A uniform class distribution can also be assumed, so we end up with the term  $Pr(d_i/c_j)$  only [13]. For a document  $d_i$ , that is composed of a sequence of words  $w_1, w_2, \dots, w_L$ ,  $Pr(d_i/c_j)$  it is expressed as follows:  $Pr(w_1, w_2, \dots, w_L/c_j)$ . We are going to write it as  $Pr_{c_j}(w_1, w_2, \dots, w_L)$  for simplicity.

$Pr_{c_j}(w_1, w_2, \dots, w_L)$  is the likelihood that  $w_1, w_2, \dots, w_L$  occurs in  $c_j$ . This can be calculated as shown in Eq. 2.

$$\begin{aligned} & Pr_{c_j}(w_1, w_2, \dots, w_{L-1}, w_L) \\ &= Pr_{c_j}(w_1) * Pr_{c_j}(w_2/w_1) \\ & \quad \dots * Pr_{c_j}(w_L/w_{L-1}, \dots, w_1) \\ &= \prod_{i=1}^L Pr_{c_j}(w_i/w_{i-1}, w_{i-2}, \dots, w_1) \end{aligned} \quad (2)$$

Nevertheless, in practice, the above dependency is relaxed and it is assumed that each word  $w_i$  is only dependent on the previous  $n - 1$  words [11]. Hence, Eq. 2 is transformed to the following equation:

$$\begin{aligned} & Pr_{c_j}(w_1, w_2, \dots, w_L) \\ &= \prod_{i=1}^L Pr_{c_j}(w_i/w_{i-1}, w_{i-2}, \dots, w_{i-n+1}) \end{aligned} \quad (3)$$

The n-gram model is the probability distribution of sequences of length  $n$ , given the training data [14]. Therefore,  $Pr_{c_j}(w_1, w_2, \dots, w_L)$  is referred to as the

n-gram language model approximation for class  $c_j$ . Now, from the training set and for each class, the n-gram probabilities are calculated using the maximum likelihood estimation (MLE) shown in Eq. 4 [15]:

$$\begin{aligned} Pr_{c_j}(wi/w_{i-n+1}^{i-1}) &= \frac{Pr(w_{i-n+1}^i)}{Pr(w_{i-n+1}^{i-1})} \\ &= \frac{count(w_{i-n+1}^i)/N_w}{count(w_{i-n+1}^{i-1})/N_w} \\ &= \frac{count(w_{i-n+1}^i)}{count(w_{i-n+1}^{i-1})} \end{aligned} \quad (4)$$

where  $N_w$  is the total number of words, and  $w_{i-n+1}^i$  is the string formed of the ‘n’ consecutive words between  $w_{i-n+1}$  and  $w_i$ . We are proposing to use the n-Gram Language model for URL-based classification. However, in our case, we will use characters instead of words as a basis of the language model. We construct a separate LM for each class of URLs as follows. The above probabilities are calculated for each class in the training set by counting the number of times all sub-strings of lengths  $n$  and  $n - 1$  occur in the member URLs of that class. For example, suppose we have the following strings as members of class  $c_j$ , {‘ABCDE’, ‘ABC’, ‘CDE’}. In a 3-gram LM, for class  $c_j$  we will store all sub-strings of length 3 and those of length 2, along with their counts, as shown in Table 1.

**Table 1.** Sample data-structure for 3-gram LM counts.

3-grams	(‘ABC’: 2), (‘BCD’: 1), (‘CDE’: 2)
2-grams	(‘AB’: 2), (‘BC’: 2), (‘CD’: 2), (‘DE’: 2)

Counts in Table 1 are acquired during the training phase. Then in the testing phase, URLs are converted into n-grams, and for each n-gram, its probability is calculated using Eq. 4. A new URL,  $URL_i$ , is classified as member of class  $c_j$ , if the language model of  $c_j$  maximizes Eq. 1, i.e. maximizes  $Pr(c_j/URL_i)$ .

### 3.1 Dealing with Unseen n-Grams

The maximum likelihood in Eq. 4 can be zero for n-grams not seen in the training set. Therefore, smoothing is used to deal with the problem by assigning non-zero counts to unseen n-grams. Laplace smoothing is one of the simplest approaches [15], calculated as follows:

$$Pr_{c_j}(wi/w_{i-n+1}^{i-1}) = \frac{count(w_{i-n+1}^i) + 1}{count(w_{i-n+1}^{i-1}) + V} \quad (5)$$

In Eq. 5, the count is increased by 1 in the numerator, and by  $V$  in the denominator, where  $V$  represents the number of unique sequences of length  $n - 1$  found in the training set. By using this, we are effectively lowering the count of the non-zero sequences and assigning a discounted value to the unseen sequences [16]. Both 1 and  $V$  can be multiplied by a coefficient  $\gamma$  in order to control the amount of the probability mass to be re-assigned to the unseen sequences. There are other more sophisticated smoothing techniques that could be applied including Witten-Bell discounting [17] and Good Turing discounting [18].

### 3.2 Linked Dependence Assumption

In the n-gram LM, in order to move from Eqs. 2 to 3, we need to assume that the probability of  $w_i$  depends only on that of the previous  $n - 1$  terms. Similarly, in the uni-gram LM, all terms are assumed to be totally independent, i.e. it is equivalent to a bag of words approach. Although, increasing the value of  $n$  relaxes the *independence assumption*, the assumption is still strong. Cooper [19], points out the *linked dependence assumption* (LDA) as a weaker alternative assumption. Lavrenko [20] explained the *linked dependence* as follows. Consider the case of a two words vocabulary,  $V = \{a, b\}$ . In the case of two classes,  $c_1$  and  $c_2$ , and under the *independence assumption*,  $Pr_{c_1}(a, b) = Pr_{c_1}(a) * Pr_{c_1}(b)$ . Similarly  $Pr_{c_2}(a, b)$  is the product of  $Pr_{c_2}(a)$  and  $Pr_{c_2}(b)$ . Otherwise, when terms are assumed to be dependent,  $Pr_{c_1}(a, b)$  and  $Pr_{c_2}(a, b)$  can be expressed as follows:

$$Pr_{c_j}(a, b) = K_{c_j} * Pr_{c_j}(a) * Pr_{c_j}(b) \quad (6)$$

where  $K_{c_j}$  measures the dependence of the terms in class  $c_j$ . Terms are positively correlated if  $K_{c_j} > 1$ , and they are negatively correlated if  $K_{c_j} < 1$ . As mentioned earlier, with the independence assumption,  $K_{c_j}$  is equal to 1. Now, in Cooper's LDA,  $K_{c_j}$  is not assumed to be equal to 1, however it is assumed to be the same for all classes, i.e.  $K_{c_1} = K_{c_2} = K_{c_j} = K$

Accordingly, the value of  $K$  might not be needed if we try to maximize the log-likelihood ratio of relevance of  $Pr(c_j/d_i)$  divided by  $Pr(\bar{c}_j/d_i)$ , rather than  $Pr(c_j/d_i)$  as in Eq. 1.  $Pr(\bar{c}_j/d_i)$  is the posterior probability of all other classes except  $c_j$ . This is similar to the approach used in the *binary independence model* (BIM) [21, 22]. Similarly, when using Language Models for spam detection, Terra created two models for ham and spam messages [23], and a message was considered to be spam if its log-likelihood odds ratio exceeded a certain ratio. Hence, the equation of our proposed classifier will look as follows.

$$\begin{aligned} \log LL_{c_j} &= \log \left( \frac{Pr(c_j/d_i)}{Pr(\bar{c}_j/d_i)} \right) \\ &= \log \left( \frac{Pr(d_i/c_j) * Pr(c_j)}{Pr(d_i/\bar{c}_j) * Pr(\bar{c}_j)} \right) \\ &= \sum_{i=1}^L \log \left( \frac{Pr_{c_j}(w_{i-n+1}^i)}{Pr_{\bar{c}_j}(w_{i-n+1}^i)} \right) + \log \left( \frac{Pr(c_j)}{Pr(\bar{c}_j)} \right) \end{aligned} \quad (7)$$

A new URL,  $URL_i$ , is classified as member of class  $c_j$ , if the language model of  $c_j$  maximizes Eq. 7, i.e. maximizes the  $\log LL_{c_j}$ . Hereafter, we refer to this variation of the n-gram LM as *Log-likelihood Odds* (LLO) model. It is worth mentioning that the use of logarithmic scale also helps in preventing decimal point overflow during the implementation.

## 4 Experiments and Datasets

After some preliminary results in [24], here we extend the experimentation to 3 datasets with different classification objectives. Our first dataset, WebKB corpus, is commonly used for web classification (e.g. [25]). It contains pages collected from the computer science departments in 4 universities. Pages are labelled according to their function in the university websites. In total, there are 7 classes-labels: course, faculty, student, project, staff, department and other. We employed the same subset of the dataset used in previous research, to be able to compare our results to them [2, 5, 26]. The subset contains 4,167 pages. Following previous researches, we used the same training and test-sets and a *leave-one-university-out cross-validation* for the WebKB URLs [2].

In addition to WebKB, we also used the categorized web pages from DMOZ, which was historically known as the Open Directory Project (ODP). This represents a problem of topic classification. Baykan et al. [5] selected 15 topics from DMOZ categories; 1,000 URLs were put aside for testing, and the remaining URLs were used to create 15 balanced training sets for their 15 binary classifiers. For the sake of useful comparison, we calculated the precision, recall and F-measure for this dataset in the same fashion as explained in [27].

Besides functional and topic classification, we also wanted to apply our approach to a different type of classification problem, namely one of classification based on language. Global Voices Online (GVO) is a website that publishes social and political articles in different languages. Thus, we created our third dataset by extracting the URLs of the most recent articles published there in 5 languages. We choose articles in 2 Latin languages (Spanish and Italian), 2 Germanic languages (Deutsch and Dutch) and articles in English. For the first 4 languages we got the URLs of the most recent 100 articles in each of them. For English articles, we got the URLs of the most recent 150 articles, in order to also test the effect of having imbalanced classes. In total we have 550 URLs. The URLs were equally split into training and test sets.

An example URL looks as follows:

*“es.globalvoicesonline.org/2013/07/08/edward-snowden-divide-a-los-rusos”*

The host part of the URL reflects the article’s language. For example, in the above URL, *es* stands for Spanish. It is then followed by the website’s domain [globalvoicesonline.org](http://globalvoicesonline.org), then the article’s data in the form of year, month and day, with forward slashes in between. The rest of the URL comes after another slash.

The final part of the URL is normally constructed from the article’s headline, however, this is not always the case. The presence of an identifier to the article’s language in the host part makes our classification problem trivial, hence we removed that part of the URLs while constructing our dataset. We also removed the domain part since it is constant in all our URLs, as well as the date part.

Hence, the example UR mentioned earlier, is being saved as follows in our dataset:

*“edward-snowden-divide-a-los-rusos”*

We are going to call the resulting dataset GVO.

The core functionality of the code used for the experiments is implemented in IRLib (Information Retrieval Library). IRLib<sup>1</sup> is written in Python and is available as Free and Open Source Software on-line.

## 5 Results

### 5.1 Results for the Primary Dataset

Kan [2] achieved an average  $F_1$  – *measure* of 22.1 % for the WebKB dataset using punctuation-based (terms) approach. The next step was to use *information content* (IC) reduction and *title token-based finite state transducer* (FST) to further segment URL terms and expand abbreviations. This achieved an average  $F_1$  – *measure* of 33.2 % and 34 % respectively. For the same dataset, the proposed n-gram LM classifier achieved an average  $F_1$  – *measure* of 51.4 %, where  $n = 4$  and  $\gamma = 0.0062$ . The *log-likelihood odds* (LLO) variation of the same LM increased the average  $F_1$  – *measure* to 59.25 %. Detailed results are shown in Table 2.

**Table 2.** Comparing  $F_1$  – *measure* for the WebKB dataset. Results in first 3 rows are from [2] using  $SVM^{light}$ , the last two rows are using the proposed n-gram Language Model ( $\gamma = 0.0062$ ). IC, FST and LLO stand for information content reduction, title token-based finite state transducer, and Log-likelihood Odds respectively. All  $F_1$  values are multiplied by 100.

Classifier	Course	Faculty	Project	Student	Macro avg
Terms	13.5	23.4	35.6	15.8	22.1
IC	50.2	31.8	35.0	15.7	33.2
FST	52.7	31.5	36.3	15.6	34.0
All-Grams	78	<b>75</b>	50	<b>63</b>	<b>66.5</b>
4-gram LM/LLO	<b>83.6</b>	40.2	<b>53.7</b>	59.4	59.25

<sup>1</sup> <https://github.com/gr33ndata/irlib>.

In later research, Kan [26] tried additional feature extraction methods, achieving the highest  $F_1 - measure$  of 52.5 %. For the same dataset, Baykan et al. [5,6] reported  $F_1 - measure$  of 66.5 % using the all-gram approach. It is clear that the classification performance of the n-gram LM for this dataset is better than all previous approaches except for all-grams. Nevertheless, the difference between results for all-grams and that of the n-Gram LM are not statistically significant ( $p=0.5$ ) applying a pairwise t-test with Bonferroni-Holm adjustment. Furthermore, it is worth noting that the n-gram LM uses only 4-grams and requires about 0.04 % of the storage and memory needed for the all-grams approach. More discussion on the scalability of the n-gram LM is included in Sect. 6.

## 5.2 Results for the Secondary Dataset

The results for DMOZ dataset are shown in Table 3. The best results for the n-gram LM were achieved using 7-grams and  $\gamma = 0.004$ . The results for the previous research using SVM and all-gram features (all 4,5,6,7 and 8-grams) [5], are also shown in Table 3. The performance of the n-gram LM is marginally better, however the statistical analysis of the results confirms that there is no statistical significance between the accuracy of the two approaches. Again, for some classes, the n-Gram LM requires less than 0.001 % of the memory and storage needed by the all-gram approach.

**Table 3.** Comparing the F-measure of the n-Gram LM and SVM (all-gram features) classifiers for DMOZ dataset. All  $F_1$  values are multiplied by 100.

Topic	SVM all-gram	n-Gram LM/LLO
Adult	<b>87.6%</b>	87.58 %
Arts	81.9 %	<b>82.03%</b>
Business	<b>82.9%</b>	82.71 %
Computers	82.5 %	<b>82.79%</b>
Games	<b>86.7%</b>	86.43 %
Health	82.4 %	<b>82.49%</b>
Home	81 %	<b>81.13%</b>
Kids	80 %	<b>81.09%</b>
News	<b>80.1%</b>	79.01 %
Recreation	79.7 %	<b>80.22%</b>
Reference	<b>84.4%</b>	83.37 %
Science	80.1 %	<b>82.52%</b>
Shopping	<b>83.1%</b>	82.48 %
Society	80.2 %	<b>81.66%</b>
Sports	84 %	<b>85.30%</b>
Average	82.44 %	<b>82.72%</b>



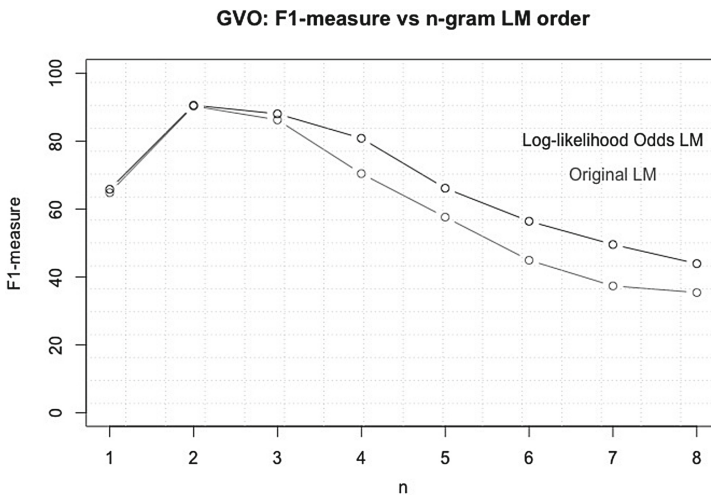
### 5.3 Results for the Tertiary Dataset

Figure 1 shows the variation of the F-measure with the value of  $n$  the in  $n$ -gram LM. It is clear that the best results were achieved for bi-grams. Thus, we compared our proposed 2-gram LM with SVM and Naive Bayes classifiers using bi-grams as their feature-sets. Table 4 shows the classification performance for the different classifiers. Applying a pairwise t-test with Bonferroni-Holm adjustment shows that the performance of the LLO variation of the 2-gram LM is significantly better than SVM ( $p = 0.004$ ) and is marginally better than NB ( $p = 0.054$ ).

We believe the reason the classification performance was much better for the GVO dataset compared to the other 2 datasets, even for a small training-set, is due to the nature of the classification problem under scrutiny. Usually, the most common  $n$ -grams in a document are the ones correlated with the language of the document. Table 5 shows the top 20 terms seen in the GVO dataset. It is clear

**Table 4.** Comparing  $F_1$  – measure for 2-gram LM and the Log-likelihood Odds (LLO) variation of the 2-gram LM with SVM and Naive Bayes (NB).  $\gamma = 0.5$  for all  $n$ -gram LMs.  $F_1$  values are multiplied by 100 (GVO Dataset).

Classifier	DE	EN	ES	IT	NL	Macro avg.
2-gram SVM (Poly. Kernel)	52.5	60.6	51.7	69.1	70.2	60.7
2-gram NB (Multinomial)	84.6	78.4	75.5	73.6	91.3	80.4
2-gram LM	87.5	88.2	92.2	92.9	88.9	89.93
2-gram LM/LLO	90.3	89.0	91.3	92.9	89.8	90.67



**Fig. 1.** The variation of the classification performance for GVO dataset, for different values of  $n$ . The value of  $\gamma$  is set to 1.

**Table 5.** The top 20 most frequent terms in GVO dataset, compared to the top 20 bi-grams there. Terms which have parts of them appearing in the top 20 bi-grams are listed in bold letters.

Terms	'die', 'il', 'di', 'of', ' <b>per</b> ', ' <b>china</b> ', 'i', 'op', ' <b>de</b> ', ' <b>en</b> ', ' <b>und</b> ', ' <b>van</b> ', 'the', 'to', 'a', ' <b>video</b> ', ' <b>del</b> ', ' <b>in</b> ', 'y', ' <b>la</b> '
2-grams	'an', 'al', 'on', 'la', 'ti', 'de', 're', 'ta', 'nt', 'or', 'in', 'si', 'di', 'ra', 'te', 'en', 'nd', 'st', 'er', 'es'

that most of them are stop words used in the 5 different languages. Stop words are normally removed in topic classification tasks since they are not correlated with specific topics. For language classification, however, as each language has its own set of stop words they are very useful. In their use of n-grams for text categorization, Cavnar et al. [28] noted that the top 300 n-grams are highly correlated with the documents' languages. Then, the less common n-grams are correlated with the documents' topics. In other words, the majority of the top 300 n-grams are common in documents of the same language, even when the documents' topics are different. We had similar findings to [28]. In combination with that, the low order of  $n$  enables the small training-set to cover a high percentage of the total bi-gram vocabulary.

#### 5.4 n-Gram LM Parameter Experimentation

Two main parameters play an important role in our n-gram LM results:

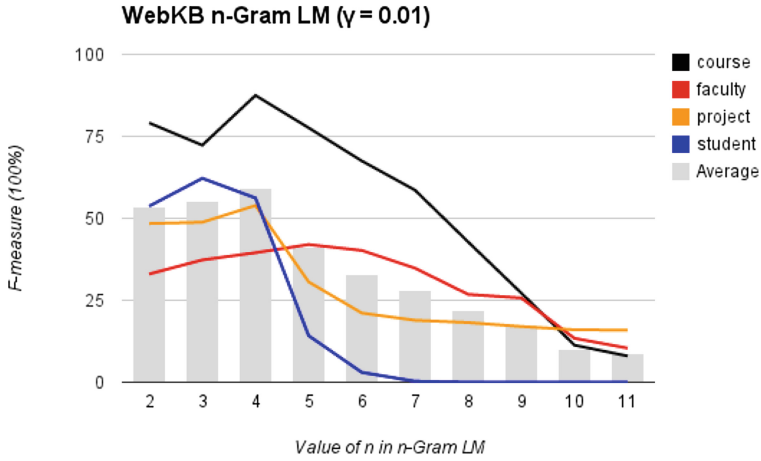
1. The order of  $n$  in the n-gram LM.
2. The value of  $\gamma$  in Laplace smoothing.

There is a trade-off between smaller and larger values of  $n$ . Higher values of  $n$  imply more scarce data and a higher number of n-grams in the testing phase that have not been seen during the training phase. On the other hand, for a lower value of  $n$ , it is harder for the model to capture the character dependencies [11]. The quantity of unseen n-grams in the testing phase is also dependent on the class distributions and the homogeneity of the class vocabularies. Classes with more samples have more chance to cover more n-gram vocabulary.

In this context, smoothing is needed to estimate the likelihood of unseen n-grams. The value of  $\gamma$  controls the amount of probability mass that is to be discounted from seen n-grams and re-assigned to the unseen ones. The higher the value of  $\gamma$  the higher the probability mass being assigned to unseen n-grams.

Figure 2 shows the variation of the F-measure with the value of  $n$  in the n-gram LM for the different class labels in the WebKB dataset. The macro-average F-measure is also shown. It is clear that the best results are achieved at  $n=4$ .

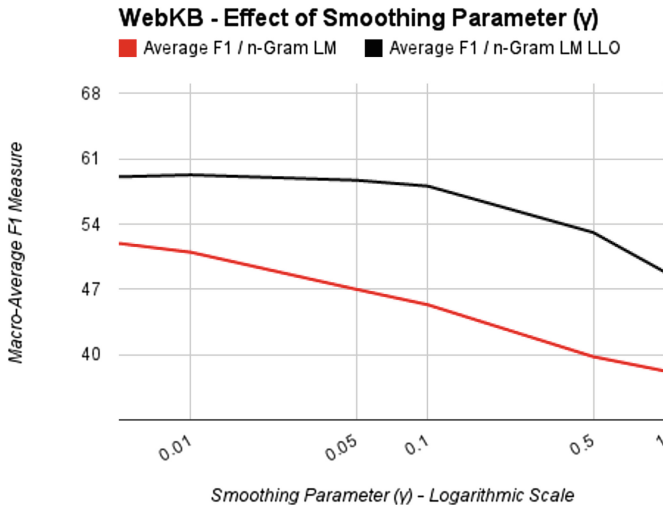
Similarly, the effect of the smoothing parameter ( $\gamma$ ) is shown in Fig. 3. Figure 3 also shows that relaxing the model's *independence assumption*, by using the *Log-likelihood Odds* model, results in better performance, and more immunity to the variations of the smoothing parameter.



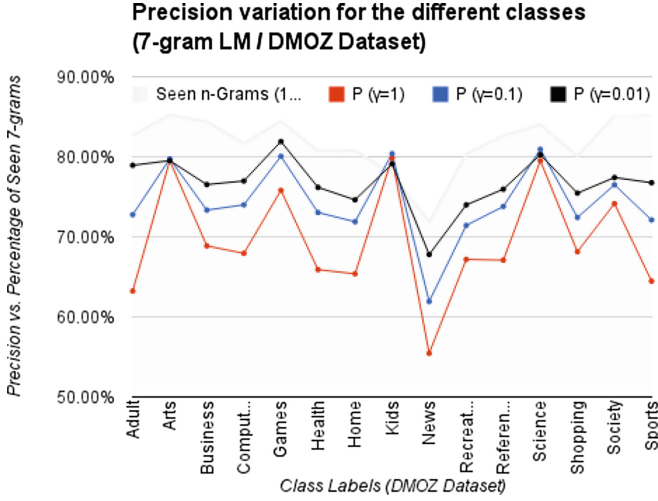
**Fig. 2.** Variations of F-measures with  $n$ .

When the model encounters a high percentage of  $n$ -grams that were never seen during the training phase, the precision of the model is affected. Smoothing, on the other hand, tries to compensate this effect by moving some of the probability mass to the unseen  $n$ -grams. As stated earlier, the amount of the probability mass assigned to the unseen  $n$ -grams is controlled by the value of  $\gamma$ .

In Fig. 4, we can see the correlation between the precision and the percentage of seen  $n$ -grams for the different classes. It is also clear that the correlation gets stronger with lower values of  $\gamma$ . For the shown models, the Pearson correlation



**Fig. 3.** Variations of F-measures with  $\gamma$ .



**Fig. 4.** Variations of Precision with classes and with the smoothing parameter,  $\gamma$ .

coefficients for the precision values with the percentages of seen n-grams are 0.51, 0.65 and 0.74 for  $\gamma = 1$ , 0.1 and 0.01 respectively.

## 6 n-Gram LM Scalability

The storage size needed for the n-gram LM is a function of the number of n-grams and classes we have, while for the all-grams approach used by Baykan et al. [5], the storage requirements are a function of the number of URLs in the training set as well as the different orders of ‘n’ used in the all-grams. This means that in the n-gram LM the memory and storage requirements can be 100,000 times less than that needed by the conventional approaches. This reduction was shown, during our tests, to also have a big impact on the classification processing time.

Let us use any of the binary-classifiers used in DMOZ dataset to explain this in more detail. We have about 100,000 URLs in the ‘Sports’ category, thus as shown in Baykan et al. [5], we will build a balanced training-set of positive and negative cases of about 200,000 URLs.

As we have seen in Eqs. 4 and 5, for an n-gram language model we need to store the counts of n-grams and (n-1)-grams for each class. Since we can achieve slightly better results than Baykan et al. [5] with ‘n = 7’, we will do our calculations based on the 7-gram LM here. The number of 7-grams in the positive and negative classes are 746,024 and 1,037,419 respectively, while the number of 6-grams for the same two classes are 568,162 and 795,192. Thus the total storage needed is the summation of the above 4 values, i.e. 3,146,797

For the approach used by Baykan et al. [5], we need to construct a matrix of all features and training-data records. The features in this case will be the all-grams, i.e. 4, 5, 6, 7 and 8-grams, and the training-data records are the 200,000

URLs in the training-set. This matrix is to be used by a Naive Bayes or SVM classifiers later on. The counts for the 4, 5, 6, 7 and 8-grams are 222,649, 684,432, 1,198,689, 1,628,422, 2,008,153 respectively. Thus, the total number of features is the summation of the above 5 values, i.e. 5,742,345. Given that there are about 200,000 URLs in the training-set, the total size of the matrix will be the product of the above 2 numbers,  $5,742,345 * 200,000$ , which is 1,148,469,000,000.

As we can see in the above example, the memory and storage requirements for the n-gram LM is 1:364,964 ( $\approx 0.0003\%$ ) of that needed for the conventional approaches. Similarly, even for a small datasets such as WebKB, the memory needed for n-gram LM is about 1:2600 ( $\approx 0.04\%$ ) of that needed for the all-grams approach.

As we have discussed earlier, such reduction in storage and processing requirements for the n-Gram LM, does not impact negatively on its classification performance compared the the previous classification approaches.

## 7 Conclusions

Here we have presented a new LM approach for URL classification that cuts down on the number of features, and therefore, the storage and processing requirements, and still manages to achieve comparable levels of performance. We have tested our approach on 3 different web page classification settings: based on function, topic and language. Our experiments show that the n-gram LM approach with very basic smoothing is offering some significant improvements for classification performance in some cases or at least equal performance over other methods such as *terms* or *all-grams* used with NB and SVM classifiers.

The n-gram LM requires less processing power compared to all-gram. For some cases the proposed model required less that 0.001% of the storage and processing power needed by the previous methods.

Our method has application to real world URL classification, an important emerging problem. We have tested it on a large dataset (some classes of DMOZ dataset have more that 200,000 URLs) as well as on the WebKB dataset. We have also performed parameter experimentation to establish the importance of parameters in the new LM.

As further work, we believe that more sophisticated smoothing methods and interpolating multiple n-gram models, with different values of n, could improve the performance of the LM model. Thus, we propose to continue our research in that direction.

## References

1. Thomas, K., Grier, C., Ma, J., Paxson, V., Song, D.: Design and evaluation of a real-time URL spam filtering service. In: 2011 IEEE Symposium on Security and Privacy (SP), pp. 447–462. IEEE (2011)
2. Kan, M.: Web page classification without the web page. In: Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters, pp. 262–263. ACM (2004)

3. Vonitsanou, M., Kozanidis, L., Stamou, S.: Keywords identification within greek URLs. *Polibits* **43**, 75–80 (2011)
4. Nicolov, N., Salvetti, F.: Efficient spam analysis for Weblogs through URL segmentation. *Amsterdam Stud. Theory History Linguist. Sci. Ser. 4* **292**, 125 (2007)
5. Baykan, E., Henzinger, M., Marian, L., Weber, I.: Purely URL-based topic classification. In: *Proceedings of the 18th International Conference on World Wide Web*, pp. 1109–1110. ACM (2009)
6. Baykan, E., Marian, L., Henzinger, M., Weber, I.: A comprehensive study of features and algorithms for URL-based topic classification. *ACM Trans. Web (TWEB)* **5**, 15 (2011)
7. Baykan, E., Henzinger, M., Weber, I.: A comprehensive study of techniques for URL-based web page language classification. *ACM Trans. Web (TWEB)* **7**, 3 (2013)
8. Chung, Y., Toyoda, M., Kitsugeregawa, M.: Topic classification of spam host based on URLs. In: *Proceedings of the Forum on Data Engineering and Information Management (DEIM)* (2010)
9. Ma, J., Saul, L.K., Savage, S., Voelker, G.M.: Identifying suspicious URLs: an application of large-scale online learning. In: *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 681–688. ACM (2009)
10. Zhao, P., Hoi, S.C.: Cost-sensitive online active learning with application to malicious URL detection. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 919–927. ACM (2013)
11. Peng, F., Huang, X., Schuurmans, D., Wang, S.: Text classification in asian languages without word segmentation. In: *Proceedings of the Sixth International Workshop on Information Retrieval with Asian Languages*, vol. 11, pp. 41–48. Association for Computational Linguistics (2003)
12. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to Ad Hoc information retrieval. In: *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 334–342. ACM (2001)
13. Grau, S., Sanchis, E., Castro, M.J., Vilar, D.: Dialogue act classification using a Bayesian approach. In: *9th Conference Speech and Computer* (2004)
14. Manning, C.D., Schütze, H.: *Foundations of Statistical Natural Language Processing*, vol. 999. MIT Press, Cambridge (1999)
15. Chen, S.F., Goodman, J.: An empirical study of smoothing techniques for language modeling. In: *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pp. 310–318. Association for Computational Linguistics (1996)
16. Jurafsky, D., Martin, J.: *Speech & Language Processing*. Pearson Education India, New Delhi (2000)
17. Witten, I.H., Bell, T.C.: The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression. *IEEE Trans. Inf. Theory* **37**, 1085–1094 (1991)
18. Good, I.J.: The population frequencies of species and the estimation of population parameters. *Biometrika* **40**, 237–264 (1953)
19. Cooper, W.S.: Some inconsistencies and misidentified modeling assumptions in probabilistic information retrieval. *ACM Trans. Inf. Syst. (TOIS)* **13**, 100–111 (1995)
20. Lavrenko, V.: *A Generative Theory of Relevance*, vol. 26. Springer, New York (2009)
21. Robertson, S.E., Jones, K.S.: Relevance weighting of search terms. *J. Am. Soc. Inf. Sci.* **27**, 129–146 (1976)

22. Jones, Sk, Walker, S., Robertson, S.E.: A probabilistic model of information retrieval: development and comparative experiments: Part 1. *Inf. Process. Manag.* **36**, 779–808 (2000)
23. Terra, E.: Simple language models for spam detection. In: *TREC* (2005)
24. Abdallah, T.A., De la Iglesia, B.: URL-based web page classification - a new method for URL-based web page classification using n-gram language models. In: *International Conference on Knowledge Discovery and Information Retrieval (KDIR 2014)* (2014)
25. Slattery, S., Craven, M.: Combining statistical and relational methods for learning in hypertext domains. In: Page, David L. (ed.) *ILP 1998. LNCS*, vol. 1446, pp. 38–52. Springer, Heidelberg (1998)
26. Kan, M., Thi, H.: Fast webpage classification using URL features. In: *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pp. 325–326. ACM (2005)
27. Baykan, E., Henzinger, M., Weber, I.: Web page language identification based on URLs. *Proc. VLDB Endowment* **1**, 176–187 (2008)
28. Cavnar, W.B., Trenkle, J.M., et al.: N-gram-based text categorization, pp. 161–175. *Ann Arbor MI* 48113 (1994)

Knowledge Discovery, Knowledge Engineering and  
Knowledge Management

6th International Joint Conference, IC3K 2014, Rome,  
Italy, October 21-24, 2014, Revised Selected Papers

Fred, A.; Dietz, J.L.G.; Aveiro, D.; Liu, K.; Filipe, J. (Eds.)

2015, XX, 622 p. 174 illus. in color., Softcover

ISBN: 978-3-319-25839-3