ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/220593795

A Comprehensive Study of Features and Algorithms for URL-Based Topic Classification

Article *in* ACM Transactions on the Web · July 2011

DOI: 10.1145/1993053.1993057 · Source: DBLP

CITATIONS
48

READS
943

4 authors, including:

Monika Henzinger
University of Vienna
279 PUBLICATIONS **11,288** CITATIONS
SEE PROFILE

Ludmila Marian
CERN
8 PUBLICATIONS **153** CITATIONS
SEE PROFILE

Ingmar Weber
Qatar Computing Research Institute
243 PUBLICATIONS **5,229** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Just-in-Time Healthy Eating Intervention View project

Phylogenetic Diversity View project

All content following this page was uploaded by Ingmar Weber on 19 August 2014.

The user has requested enhancement of the downloaded file.

A Comprehensive Study of Techniques for URL-Based Web Page Language Classification

EDA BAYKAN, Izmir University
MONIKA HENZINGER, University of Vienna
INGMAR WEBER, Yahoo! Research Barcelona

Given only the URL of a Web page, can we identify its language? In this article we examine this question. URL-based language classification is useful when the content of the Web page is not available or downloading the content is a waste of bandwidth and time.

We built URL-based language classifiers for English, German, French, Spanish, and Italian by applying a variety of algorithms and features. As algorithms we used machine learning algorithms which are widely applied for text classification and state-of-art algorithms for language identification of text. As features we used words, various sized n-grams, and custom-made features (our novel feature set). We compared our approaches with two baseline methods, namely classification by country code top-level domains and classification by IP addresses of the hosting Web servers.

We trained and tested our classifiers in a 10-fold cross-validation setup on a dataset obtained from the Open Directory Project and from querying a commercial search engine. We obtained the lowest F1-measure for English (94) and the highest F1-measure for German (98) with the best performing classifiers.

We also evaluated the performance of our methods: (i) on a set of Web pages written in Adobe Flash and (ii) as part of a language-focused crawler. In the first case, the content of the Web page is hard to extract and in the second page downloading pages of the "wrong" language constitutes a waste of bandwidth. In both settings the best classifiers have a high accuracy with an F1-measure between 95 (for English) and 98 (for Italian) for the Adobe Flash pages and a precision between 90 (for Italian) and 97 (for French) for the language-focused crawler.

Categories and Subject Descriptors: H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Document and text processing, Web page classification, language classification, URL

ACM Reference Format:
Baykan, E., Henzinger, M., and Weber, I. 2013. A comprehensive study of techniques for URL-based web page language classification. *ACM Trans. Web* 7, 1, Article 3 (March 2013), 37 pages.
DOI: http://dx.doi.org/10.1145/2435215.2435218

1. INTRODUCTION

The task of language classification is to determine the language in which a piece of text is written. In this article, we study URL-based Web page language classification, that

Preliminary results of this study are published in Baykan et al. [2008].

Authors' addresses: E. Baykan (corresponding author), Software Engineering Department, Izmir University, Turkey; email: eda.baykan@gmail.com; M. Henzinger, Department of Computer Science, University of Vienna, Austria; I. Weber, Yahoo! Research Barcelona, Spain.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2013 ACM 1559-1131/2013/03-ART3 $15.00
DOI: http://dx.doi.org/10.1145/2435215.2435218

is, the problem of determining the language of a Web page using only its URL and *not* its content or its incoming hyperlinks. We study this problem for the main European languages: English, German, French, Spanish, and Italian.

There are various applications for URL-based Web page language classification. Its main applications are classifying Web pages that have only multimedia content and improved language-aware crawling in Web search engines. For multimedia pages it is often difficult or impossible to automatically obtain a textual representation of the content which could then be used for language classification. Although commercial search engines have proprietary techniques to (attempt to) extract text from Web pages written in Adobe Flash to include them in their indexes, we still view this file type as representative or at least indicative of content where such extraction is infeasible. Crawlers of major Web search engines attempt to satisfy a certain download quota for languages and maintain a list of URLs of all uncrawled Web pages. They can avoid downloading Web pages in a language whose download quota is already fulfilled by using the predictions of our classifiers to prioritize the URLs to be crawled next. Such an approach could reduce bandwidth requirements and the crawling would be faster as the unnecessary downloads can be avoided. A special case of this scenario are crawlers for language-specific search engines, such as `www.yandex.ru` or `www.voila.fr`. Other potential applications for URL-based language classification are regrouping/filtering the results of Web search by language and showing language-related icons when the user is hovering with the mouse over the hyperlinks on the Web page.

To build a URL-based language classification system we first experimented with two baseline algorithms which do not require any a priori language models for classification. The first baseline algorithm is based on the assumption that the language of a Web page can be easily determined using the country code top-level domain, such as `.de` or `.fr`. Our experiments show that this assumption does not hold because of the heterogeneous nature of the largest two top level domains. According to large-scale studies Baykan et al. [2006, 2009a], about 60% of Web pages belong to the `.com` domain and about 10% belong to the `.org` domain. Language classifiers which only use the country codes generally fail to classify the Web pages in these top-level domains since these Web pages do not have top-level domains specific to a country. Assigning these two domains to English creates many false assignments. The second baseline algorithm, which is based on IP addresses, assigns the language of a Web page to the language spoken in the country where the server hosting the Web page is located. This algorithm gave more promising results than the first baseline algorithm, however, it fails to classify French, Spanish, and Italian Web pages which are hosted at Web servers located in English- or German-speaking countries.

Next we wanted to explore whether the handicaps of the aforementioned baseline algorithms can be overcome by machine learning approaches. Thus we built a training and various test sets and experimented with state-of-the-art algorithms for text classification and language identification in combination with various features like words, n-grams, and custom-made features. Specifically, we used as machine learning algorithms Support Vector Machines (SVM), Decision Trees, Naïve Bayes, and Maximum Entropy. We also applied similarity-based algorithms like Rank Order Statistics [Cavnar and Trenkle 1994], Markov [Dunning 1994], and Relative Entropy [Sibun and Reynar 1996], which are widely used for language identification. In addition to experimenting with various algorithms and features to obtain high-performance URL-based language classifiers, we also analyzed the impact of using various feature sets (e.g., varying the size of character n-grams) and the impact of varying training size on the classifier performance. We also proposed a novel feature set for representing URLs. Our custom-made feature set consists of features extracted from URLs such as the

count of occurrences of words in various dictionaries (thesaurus, city), the language assigned by an IP-based baseline algorithm, and features indicating whether the Top-Level Domain (TLD) of a URL belong to TLD lists of languages we study. Furthermore to improve the performance even more we formed new classifiers by combining the IP-address-based algorithm with a classifier that uses Support Vector Machines as algorithm and character n-grams as features.

We built our test and training sets as follows. We obtained URLs for English, German, Italian, French, and Spanish by downloading the Open directory project [1] and querying a commercial search engine with certain language-specific queries. We used this dataset (ODP + SER) to train and test our classifiers in a 10-fold cross-validation setup. Additionally, we evaluated the performance of our classifiers by classifying the language of Adobe Flash pages and Web crawl pages. The latter two test sets were build independently of the test dataset and, thus, their results are indicative of the performance of the different techniques in a production setting.

Our contributions in this article are as follows: (1) We present a comprehensive experimental evaluation of features and algorithms. (2) We show that baseline algorithms using ccTLD and IP are not enough for high performance, though they achieve high precision. (3) We show that it is possible to build high-quality URL-based language classifiers. Specifically using the SVM algorithm in combination with multiple character n-grams we achieved an F1-measure between 94 (for English) and 97 (for German). (4) We propose a novel feature set, custom-made features, that consists of information such as IP addresses, ccTLD, and number of words in various dictionaries. Using a Decision Tree algorithm with custom-made features leads to an F1-measure between 92 (for Spanish) and 98 (for German) on the ODP + SER dataset. (5) We explain why character n-grams and custom-made features have more information than words features to classify URLs. (6) We propose a novel classifier that is a combination of the IP algorithm with SVM. It improves the F1-measure by 1–2 points for all languages except for English and German. (7) We present a comparison of the best performing classifiers along various additional dimensions, such as the impact of training size. (8) We show that language classification of Web pages that have multimedia content and language-focused crawlers can benefit from URL-based language classifiers as we obtain high classification performance on these use-cases with our best performing classifiers. For example, on the multimedia dataset, which is only used for testing purposes and *not* for training, our best approach obtains F1-measures of at least 95 points.

The rest of this work is organized as follows. In Section 2 we describe the techniques we applied for the URL-based language classification problem, covering different feature sets and different algorithms. In Section 3 we present the experimental setup, including details of the datasets and the evaluation measures used. In Section 4 we present the performance of our URL-based language classifiers. First we discuss the performance of baseline algorithms then we give the performance of the machine learning and similarity-based algorithms with an emphasis on the features they are used with. Finally we give a comparison of the best performing classifiers and how their performance changes when the training size varies. In Section 5 we apply our URL-based language classifiers trained on the ODP-SER dataset to identify the language of Adobe Flash Web pages and the Web crawl pages. In Section 6 we discuss work related to language classification in general and related to Web page language classification in particular. Finally, we summarize our main results in Section 7.

[1] `http://rdf.dmoz.org/`

2. LANGUAGE IDENTIFICATION BASED ON URLS

In this section we explain the features and the algorithms used in our URL-based language classifiers. To represent URLs in classifiers, one has to map them to numerical feature vectors. We discuss different ways of doing this in Section 2.1. In our URL-based language classifiers we used two baseline algorithms, a variety of machine learning algorithms, and the state-of-the-art algorithms for language identification. We explain these algorithms in detail in Section 2.2.

2.1. Mapping URLs to Features

We experimented with three different methods to extract features from URLs: using words as features, using n-grams as features, and using our novel custom-made features. Words and n-grams are commonly used features in the language identification literature [Cavnar and Trenkle 1994; Dunning 1994; Grefenstette 1995; Ingle 1976; Rhekurek and Kolkus 2009; Sibun and Reynar 1996]. Custom-made features comprise various features such as the country code top-level domain (ccTLD), host-IP-derived country estimate, and the number of words in a dictionary.

Words as features. Each URL is split into a sequence of strings of letters at any punctuation marks, numbers, or other nonletter characters. Resulting strings of length less than 2 and the following special words are removed: "www", "index", "html", "htm", "http", and "https". We refer to a single valid string as a *token*. For example, `http://vldb.org/vldb_journal/` would be split into the tokens `vldb`, `org`, and `journal`. Classification algorithms using words features keep counters for the number of times a certain token is seen in the URLs of a given language. This way algorithms can learn that tokens such as `cnn` or `gov` are indicative of English, whereas `produits` or `recherche` are indicative of French.

N-grams as features. N-gram is a subsequence of n letters from a given sequence of letters. The main advantage of n-grams over tokens is the capability to detect subwords without requiring an explicit list of valid terms. We experimented with various single n-gram sizes ranging from 1-gram (individual characters) to 7-grams. 1- and 2-grams did not perform very well and 7-grams were already very similar to using whole tokens and so we did not try bigger n-grams. We refer to the (combination) of 3-4-5-6-7-grams as *allgrams*.

There are two ways to extract n-grams from URLs. (1) Tokens are obtained from the URL and n-grams are extracted from tokens. N-grams derived with this approach are named as grams-fromWord. (2) N-grams are obtained directly from URL. N-grams derived with this approach are named as grams-from-URL.

For example, to form 3-grams-from-Word, a URL is first split into tokens. Then 3-grams, that is, sequences of exactly three letters, are derived from them where special characters are used to mark the beginning and the end of a token. As an example, the token `weather` gives rise to the trigrams "_we", "wea", "eat", "ath", "the", "her", and "er_". A possible advantage of using n-grams over using only full words is that n-grams learn characteristics of a language, for example, the 3-grams "_th" or "ing" are very common in English. 3-grams-from-URL would generate the trigram "yde" for the URL `http://www.hi-fly.de`, which we would not be generated with 3-grams-from-Word .

Custom-made features. We created a fixed number of special features for each URL. These features are mostly derived from dictionaries, from information concerning the

Table I. ccTLD to Language Mapping

Language	ccTLD List
English	au, ca, cg, edu, gb, gh, gov, ie, ke, mil, mw, ng, nz, sd, tz, ug, uk, um, us, za, zm
German	at, ch, de, li, lu
French	bf, ci, cm, dz, fr, gf, gn, ht, mg, ml, ne, pf, sn, td, tf, tn
Spanish	ar, bo, cl, co, cu, ec, es, gt, mx, pe, pr, ve
Italian	it, va

Table II. The Size of Dictionaries Used for Custom-Made Features

Language	Dictionaries		
	OpenOffice	City	Trained
English	60,306	473	4,512
German	85,892	1,640	5,009
French	62,210	512	5,177
Spanish	68,406	146	5,299
Italian	95,000	484	4,838

top-level domains, and from information based on the IP addresses of the servers hosting the Web pages.

— *Top-level domain country code.* For each language we used a small number of Top-Level Domain (TLD) country codes. For example, we counted ".us", ".uk", and ".nz" as top-level domains for English. Each URL has features in the custom-made feature vector which indicate whether the TLD of the URL belongs to TLD lists of the languages we study. The full list of top-level domains for languages is given in Table I. We also used binary features for the occurrences of country codes in the other parts of the URL. For these generalized features a URL such as `http://fr.search.yahoo.com/` would have 1 for the "fr" feature before the first directory sign in the URL.
— *Other top-level domains.* We used three binary features (separately) to keep track of whether a URL is in the ".net", in the ".org", and in the ".com" domain.
— *OpenOffice dictionaries.* We counted the number of tokens present in OpenOffice dictionaries[2] which were downloaded from wiki services [3]. In Table II we present the sizes of OpenOffice dictionaries. Note that during the construction of the dictionaries no effort was made to detect compound words such as cheapflights.
— *City dictionaries.* We used lists from Wikipedia to construct a dictionary of cities for each language. This way we can, for example, tell that Berlin is a city in a German-speaking country. We added these lists because the OpenOffice dictionaries tend to have large cities (Paris, London, Berlin, etc.) in all the languages but miss smaller towns. In Table II for each language we present the sizes of dictionaries with city names.
— *Trained dictionaries.* We also trained dictionaries on all the URLs in the training set we used to train our classification system. We will explain our training sets with detail in Section 3.1. The words in the trained dictionaries are selected among the words extracted from the URLs in the training set. The selection procedure to add a word or token to the trained dictionary of a language is as follows. A token is added

[2]We used the following spelling dictionaries. English: United States. German: Germany, by F. M. Baumann. French: France Classique. Spanish: Spain-etal. Italian: Dizionario Italiano.
[3]`http://wiki.services.openoffice.org/wiki/Dictionaries`

to the dictionary for a language X if this token: (i) appeared in at least .01% of the training URLs of language X, and (ii) at least 80% of the training URLs in which the token appeared belonged to language X. This way, for example, the token "cheap" gets added to the English trained dictionary and the token "voyageurs" gets added to the not-English trained dictionary. Only tokens of minimum length of 3 were included in the dictionary. Note that the words as features setting discussed before can be seen as implicitly using a more fine-grained version of the trained dictionary. In the last column of Table II we present the sum of the sizes of trained dictionaries (i.e., sum of the size of English trained dictionary and not-English trained dictionary) for each language classifier.

— *Number of hyphens.* Preliminary experiments showed that, somewhat surprisingly, hyphens occur about 2 times more often in German URLs than in English URLs. That is why we included this counter in the feature set.

— *IP.* This feature is based on the IP address for the hostname extracted from a URL. It indicates whether the location country of the server, as estimated using a standard IP geo-location library, has one of the languages we study as official language. For example, the feature "English IP" refers to the feature indicating whether the URL is hosted at a Web server which is located in an English-speaking country.

In total, including small variants of features discussed previously and keeping counters separately before the first "/" of a URL and after, we obtained 60 custom-made features for each URL.

2.2. Classification Algorithms

Our baseline algorithms for language classification use the ccTLD and the hostname of the URL. For automated text classification, machine learning algorithms like Support Vector Machine, Decision Tree, Naïve Bayes, and Maximum Entropy are widely used [Sebastiani 2002]. Thus we used these machine learning algorithms in our experiments. We refer the reader to text books, such as by Hastie et al. [2001], for a detailed explanation of the machine learning algorithms. Commonly used algorithms [Cavnar and Trenkle 1994; Dunning 1994; Sibun and Reynar 1996] for language identification build language models and classify documents with the label of the most similar language model, each using different metrics for similarity. We applied these similarity-based algorithms for the URL-based language identification task.

We briefly describe the algorithms used in our classifiers next.

Country Code Top-Level Domain (ccTLD). This simple baseline algorithm uses only country code top-level domains (ccTLD). This algorithm will be abbreviated as *ccTLD*. The ccTLD algorithm takes the ccTLD of a URL, checks the official language for the ccTLD's country, and assigns the corresponding language to the URL. In Table I we present the mapping between ccTLDs and the languages spoken in the corresponding countries. In this table we included ISO 3166-1 codes for all the countries where English, Spanish, German, French, or Italian was an official language and where the country's population was at least 10M. When there were several languages spoken in a country, they are ranked in order of prevalence as judged by reading the related Wikipedia article and the most spoken language is chosen as the language spoken in that country. For example, we assigned German as the official language of Switzerland and English as the official language of Canada.

We can count some example mappings that ccTLD algorithms make between languages and ccTLDs as follows. For French it uses the ccTLDs `fr` (France), `tn` (Tunisia), `dz` (Algeria), and `mg` (Madagascar). For German it uses `de` (Germany) and `at` (Austria). For Italian it uses `it` (Italy) and `va` (Vatican City). For Spanish it uses `es` (Spain), `cl` (Chile), `mx` (Mexico), `ar` (Argentina), `co` (Colombia), `pe` (Peru), and `ve` (Venezuela). For

English it uses `au` (Australia), `ie` (Ireland), `nz` (New Zealand), `us`, `gov`, `mil` (United States), and `gb` and `uk` (United Kingdom).

Country code top-level domain plus .com and .org (ccTLD+). The ccTLD+ algorithm is essentially the same as ccTLD algorithm, the only difference being that we add the `.com` and `.org` TLDs to the set of English TLDs. This variant will be abbreviated as *ccTLD+*. Both ccTLD and ccTLD+ only work with the trivial set of features and can obviously not be applied to other feature sets.

IP. The main idea of IP-based Web page language classification is to assign the language of a Web page to the language spoken in the country where the server hosting the Web page is located. This algorithm is abbreviated as *IP* and uses the hostname extracted from URL as feature. The hostname is then mapped to an IP address via a DNS lookup[4]. In order to map the IP address of the Web server to the country where it is located, we used MaxMind GeoLite Country API [5]. Then we did mapping between the country and the language spoken in that country by using Table I. Interesting questions arising with IP-based classification are, for example, "Do Web servers in Italy host mostly Italian pages?" and "Are all Italian Web pages hosted by servers in Italian-speaking countries?". Related questions were investigated in Cambazoglu et al. [2010] where the authors describe algorithms to: (i) partition a Web search index for geographically distributed search engines, and to (ii) adaptively forward queries to other partitions of the index. The language and country of a Web page were two of the features they used in making the corresponding decisions.

Support Vector Machine (SVM). This algorithm is a linear learning system that builds two-class classifiers. The underlying idea is to map the training points to a higher-dimensional vector space and then to find a hyperplane separating most of the positive and negative training points. For efficiency the algorithm avoids ever computing this mapping explicitly using kernel functions. For the implementation of Support Vector Machine algorithm [Tsochantaridis et al. 2005] we used SVM^{perf} [Joachims 2009] software package with its default settings. For words and n-grams we used a standard tf-idf weighting[6] of features.

Naïve Bayes (NB). This algorithm assumes conditional statistical independence of the individual features in the test item given the class. It then applies the maximum likelihood principle to find the class which is the most likely to generate the observed feature vector of the test item. In our setting, where we want to classify Web pages, the features correspond, for example, to word counts on the Web page. The model parameters, like prior probability for the class and the frequency of a words are determined from the training data. The conditional probability of a word given a class is the relative frequency of the word in the training documents belonging to the given class. The conditional probabilities of all the words seen in a document are multiplied with the prior probability to calculate the probability of a class given a document.

Maximum Entropy (ME). The idea behind this approach is to find a distribution over the observed features which explains the observed data but at the same tries to maximize the entropy in this distribution. The word counts extracted from the labeled training data are used to set constraints on the distribution. The Improved Iterative

[4]Note that a DNS lookup is negligible, both in terms of latency and bandwidth, compared to downloading the actual content.

[5]`http://www.maxmind.com/app/geolitecountry`

[6]The exact weighting formula was $w_{i,j} = f_{i,j} \cdot \ln(n/(n_i + 1))$, where $f_{i,j}$ is the occurrence count of term/n-gram i in URL j and n_i is the number of URLs containing term/n-gram i.

Scaling approach finds the most uniform distribution while satisfying the constraints derived from training data. Details about this algorithm can be found in Nigam et al. [1999].

For Maximum Entropy and Naïve Bayes classifiers in combination with words and n-gram features we used the Bow Toolkit [McCallum 1996] with its default settings. As NB and ME are built for discrete, probabilistic models, we used the raw feature counts without any weighting. For Maximum Entropy in combination with custom-made features we used MATLABArsenal[7] and for Naïve Bayes with custom-made features we used Matlab [MathWorks 2013].

Decision Trees (DT). This algorithm builds a binary tree where the inner nodes correspond to tests on a single feature and each leaf corresponds to a classification. Decision trees have the desirable property of being easy to interpret. The decision tree is constructed greedily from the training dataset, where at each step the feature which reduces the impurity the most is added as a node. Impurity is measured with the information gain measure which is based on entropy. After the decision tree is built from the training set, the test items get classified after following the path with decision features satisfying the features in the test item. We applied decision trees only to the custom-made feature set. The reason for this is that a decision tree on word features or n-grams would give a gigantic tree, where each decision node corresponds to a particular n-gram or word, and the tree is no longer interpretable. For the implementation of DT we used the Statistical toolbox at MathWorks [2013].

Relative Entropy (RE). This algorithm learns a probability distribution for each of the languages by counting the specified features in the training set. The probability distribution for a test URL is calculated by counting the features in the test URL. To a test URL the language which minimizes the relative entropy between the probability distribution of the test URL and the probability distribution of the languages in the training set is assigned. This algorithm was proposed by Sibun and Reynar [1996].

Markov. This algorithm is proposed in Dunning [1994] and it assumes that the next character in a string only depends on a certain number of previous characters. For example, for 4-grams its previous three letters are taken into account. A score is calculated for each n-grams feature in the test URL by multiplying the count of n-grams in the test URL with the probability of generating that n-gram from its previous n-1 grams in the training language. A test URL is classified with the language which maximizes the sum of scores for its features.

Rank Order Statistics (ROS). This algorithm [Cavnar and Trenkle 1994] constructs the profiles for languages by ranking the features with the number of occurrences in the training documents. The algorithm keeps the most frequent 400 features and their ranks in the training language profile. The profile for a test document is constructed similarly. For each feature in the test URL, we calculate the difference of its rank in the test profile and its rank in the training profile. Then we sum the differences for each feature and find the distance between the test URL and the training language. A test document is classified belonging to the language having the minimum distance to the test document.

For the baseline algorithms (ccTLD, IP) and the similarity-based algorithms (RE, Markov, ROS) we implemented the algorithms in Perl. For the other algorithms we used the software packages in Joachims [2009] and McCallum [1996].

[7] `http://www.informedia.cs.cmu.edu/yanrong/MATLABArsenal/MATLABArsenal.htm`

3. EXPERIMENTAL SETUP

In order to train and test the feature sets and the algorithms used in our classifiers we used various datasets. Baseline algorithms ccTLD and IP have the nice property that they do not require *any* labeled training URLs. But the other algorithms need labeled training data to form the language models in the classifiers.

In Section 3.1 we first describe these datasets, then in Section 3.2 we discuss how we divided them into training and test datasets. Finally in Section 3.3 we explain the evaluation measures used to evaluate the performance of our classifiers.

3.1. Datasets

URLs labeled with their true language are necessary to train and test our classifiers. We downloaded Open Directory Project and queried a search engine in various ways to obtain English, German, Italian, French, and Spanish URLs.

Open Directory Project (ODP). In late September 2010 we downloaded the rdf dump[8] for the Open Directory Project which is the biggest human-edited directory of the Web. Volunteers assign Web pages to directories and ODP lists the URLs of these Web pages under the assigned directories. To obtain English URLs we used the URLs under all the directories in ODP except for World and Regional. For German, Italian, French, and Spanish we used the URLs listed under the corresponding language directories of the World directory, that is, World/German/. Our URL set included a total of 2.4M URLs, of which 1.3M were English and only 150k were Spanish. To have an equal number of URLs available for each language, we sampled uniformly at random 150k URLs from each of the five languages[9].

We took the language assigned in ODP as a ground truth except for one case. In ODP URLs are listed under multiple language directories due to a specific policy which we will explain with an example case. A Web page is multilingual if it has its entire content in various languages. For example, `http://luckyjob.eu` is a Web page where job offers are announced and its whole content is only in English. There are other versions of this page written in different languages. The URL of the French version of this Web page is `http://luckyjob.eu/?lang=fr` and the URL of the Italian version is `http://luckyjob.eu/?lang=it`. On the English version of the Web page there are language flags which link to the versions in the other languages. ODP editors listed the URL for the English version of the Web page under English directory, French, and Italian directory because it was obvious to reach the French and Italian versions from the English version. For example, under the Italian directory of ODP `http://luckyjob.eu/?lang=it` was never listed instead `http://luckyjob.eu` is listed. Thus to remove false positives for languages we removed the URLs like `http://luckyjob.eu` listed under multiple language directories. This removed set was 1.7% of all the obtained URLs from ODP. However, if from the English Web page it were not obvious to trace links to the other language versions, the editors would have listed the language-specific URLs under the related language directories. We want to note that in our ODP dataset we included multilingual Web pages if they have language-specific URLs and are listed only under the related language directories. For example, `http://adrozd.free.fr/` is a URL for a French Web page and it is listed under World/French. The URL of the English version of this Web page is `http://adrozd.free.fr/english1.html` and it is listed under Science. Thus we

[8] `http://rdf.dmoz.org/`

[9] For non-Spanish languages we were able to sample 150k URLs because after removing duplicates from the URLs extracted from ODP rdf dump we still had more than 150k URLs but for Spanish we had around 144k URLs.

Table III. Lists of 10 Stop-Words Specific to Each Language

Language	Stop list
English	the, of, to, and, is, it, you, that, he, was
German	dem, sich, auf, als, auch, wird, oder, aus, wurde, werden
French	et, du, une, est, pour, qui, dans, par, plus, pas
Spanish	no, el, es, por, me, te, los, para, pero, yo
Italian	non, di, che, per, sono, ho, ma, ha, ti, cosa

To obtain these lists we consulted Wikipedia frequent word lists for languages. We filtered the words common to multiple languages. We used these lists while querying a search engine to get true labeled URLs.

Table IV. Percentage of URLs in the Datasets which Have One of the Words or TLDs Used to Query the Search Engine in Their Token Set

Language	ODP	SER	ODP + SER
English	7%	46%	22%
German	76%	86%	80%
French	38%	74%	53%
Spanish	18%	76%	42%
Italian	68%	72%	69%

included the URL for the English version in our English sample ODP set and the URL for the French version in our French sample ODP set.

Search Engine Results (SER). We used a commercial search engine to obtain roughly 100k URLs for each language. Here we used the search engine's option to limit the search scope to pages written in a particular language. However, we used one of two further restrictions to avoid that any false positives were reported by the search engine. In one setting we additionally limited the search scope to a particular ccTLD. Here we used `.uk` for English, `.de` for German, `.fr` for French, `.es` for Spanish, and `.it` for Italian. The queries themselves then consisted simply of 1 to 30 as numbers, and not written out as strings. We chose the numbers as they should appear somewhere on most pages so that no significant bias is created. In total, we obtained about 30k URLs for each language. In the second setting, we dropped the ccTLD restriction and replaced it by the requirement that certain stop-words of a language should be present. Concretely, we used lists of the most frequent words in each language to compile lists of 10 stop-words specific to each language. To obtain these lists we consulted Wikipedia frequent word lists [10]. Table III illustrates the list of stop-words used for languages. Words common to multiple lists, such as "la", were removed. We then cycled through these lists, using 8 of the 10 stop-words in combination with a number from 1 to 10 as the search engine query. In total, we obtained about 70k URLs for each language this way. No URL was returned as a result for more than one language.

One interesting issue is the impact of using language-specific words as queries on the returned URLs as search engine results. In Table IV we present the percentage of URLs which have one of the language stop-words or TLDs used to query the search engine as a token. It is observed that for each language at least 46% of the URLs in SER set contain one of the most frequently used words or TLDs related to the languages. The percentages for ODP dataset are lower. These high percentages for SER dataset may be due to the features used in search engines like the text similarity of

[10]`http://en.wiktionary.org/wiki/Wiktionary:Frequency_lists/`

URL and the search engine query. A recent patent [Poola and Ramanujapuram 2007] presents techniques for tokenizing URLs and extracting keywords from URLs which can be used by search engines.

ODP + SER. If we were able to download all the URLs for a language on the Web or a uniform random sample of it then our dataset would be perfect. However, this is not possible due to huge size of the Web and due to the biases in techniques for obtaining uniform samples [Baykan et al. 2009a]. Thus for each language we merged the corresponding URLs in ODP and SER dataset to use all the available true labeled samples. URLs contained both in ODP and SER, about .1%, were unique and only included once. URLs found for several languages, well under .01% of the final count, were removed.

The underlying problem about deciding the appropriate, best dataset with true language labels stems from the lack of a large, uniform sample of the Web [Baykan et al. 2009a]. The questions "Find a uniform random sample of English URLs" and "Find an uniform sample of the Web pages" have not been answered yet and might be impossible to answer if dynamically generated pages are taken into account. In other words, what the representative sample of English URLs or German URLs should be is not known. Thus we decided to use both ODP and SER as our datasets as well as making an effort to isolate and discuss the impact of various factors. Even if the performance numbers might change on a different dataset we believe that our general observations and trends remain valid.

For each language in ODP + SER dataset the size of true labeled samples is 250k. While presenting the performance of classifiers we focus on ODP + SER dataset.

3.2. Construction of the Training and the Test Datasets

We built binary classifiers for English, German, Italian, French, and Spanish. For example, we built one classifier for English, which classifies URLs into "English" and "Not-English", and a separate classifier for "Spanish" which classifies URLs into "Spanish" and "Not-Spanish". To train and test our binary classifiers positive and negative language sample URLs are necessary. Note that classifiers using ccTLD and IP baseline algorithms do not require any training data but the rest of the algorithms do.

For each language we obtained true labeled URLs with the approaches explained in Section 3.1. We used these true labeled URLs (250k) as a positive sample in the language classifiers. To obtain a negative sample of URLs (250k) for each language classifier, we randomly subsampled[11] the positive sample of the other four languages. At the end of this process, that is, for the English classifier we obtained 250k English URLs as a positive sample and 250k not-English URLs as a negative sample. The obtained negative sample for the English classifier included 62.5k randomly selected URLs from German, Italian, French, and Spanish positive samples.

We did 10-fold cross-validation for each language classifier on ODP + SER dataset. Thus we randomly divided the positive and the negative samples into 10 folds. Each fold has equal size, up to rounding differences, and in each fold the proportion of the positive and the negative samples is the same. In other words we did stratified 10-fold cross-validation. We did 10 iterations of training and testing phase for each classifier. In each phase one fold is used for testing and the other remaining folds are used for training. In other words 10% of the data is used for testing and 90% is used for training. In each iteration a different fold is used for testing. That is why each URL in the positive and negative sample is tested once.

[11]Using all roughly 1.25M URLs to train each binary classifier would have led to too conservative classifiers as the negative samples (1M) would have dominated.

3.3. Evaluation Measures

For each of the evaluated classifiers we report the following three numbers.

Precision P. This is the number of all URLs correctly identified as belonging to language X, divided by all URLs reported to belong to that language.

Recall R = positive success ratio $p(+|+)$. This is the number of all URLs correctly identified as belonging to language X, divided by the total number of URLs for that language. It can also be referred to as "positive success ratio", as it measures the performance on the positively labeled URLs. Note that a $p(+|+)$ of 1.0 is trivial to achieve by classifying everything as belonging to the language.

Negative success ratio $p(-|-)$. Here, we divide the number of correctly identified negative URLs by the total number of negative URLs. Similar to $p(+|+)$, a $p(-|-)$ of 1.0 is trivial to achieve by classifying every URL as negative.

Giving all three numbers is somewhat redundant as

$$P = n_+ p(+|+) / \left(n_+ p(+|+) + n_- (1 - p(-|-))\right),$$

where n_+ (n_-) is the number of positive (negative) test samples. We give the precision and recall numbers to allow for easy comparison with prior work. However, notice that the precision P can be brought arbitrarily close to 1.0 by increasing the number of positive test samples n_+ and keeping n_- fixed, and it can be brought arbitrarily close to 0.0 by increasing the number of negative test samples n_- and keeping n_+ fixed, unless the classifier has $p(-|-) = 1.0$. Thus we think that $p(+|+)$ (= recall) and $p(-|-)$ are better metrics to use and, hence, we present $p(-|-)$ in addition.

In scenarios with a strong bias between the languages, the classifier for the biggest language, the case in Section 5 (with biggest n_+ and hence smallest n_-) would automatically have a larger precision value, although both its $p(+|+)$ and $p(-|-)$ might be lower than the classifier for another language. To avoid this problem we report precision P always for a balanced setting with $n_+ = n_-$. To obtain these numbers we first compute the $p(+|+)$ and $p(-|-)$ on all the URLs in our test set, and then use the preceding formula to compute the P for the balanced setting. Although the P is thus in fact redundant, we still chose to include it in our tables, as it is traditionally given in the literature. Note that our procedure for computing P gives us the true limit, which we would obtain if we took infinitely many equally sized positive and negative test samples. Any imbalance among the languages for the negative samples, for the test dataset in Section 5.1, is fully preserved in this procedure.

Our main evaluation metric is the F1-measure, defined as the harmonic mean of precision and recall. It is sometimes in the literature also referred to as balanced F-score, as it is just one representative of the family $F_\beta = (1 + \beta^2)(P \cdot R)/(\beta^2 \cdot P + R)$. Note that in our setting an F1-measure of 67 is trivially achieved by always outputting "yes", corresponding to $R = 100$ and $P = 50$.

To calculate the performance of a classification system across a number of test instances there are two widely used notions of "averages". (1) *Macro-averaged* recall, precision, and F-measure are calculated by first calculating recall, precision, and F1-measure for each class and then taking the average of the calculated values for each class to form the macro-averaged recall, precision, and F1-measure. (2) *Micro-averaged* recall, precision, and F1-measure are calculated by calculating recall, precision, and F1-measure for all the test documents in the system. In this article while presenting averaged results for F1-measure, recall, and precision we prefer to give macro-averaged values instead of micro-averaged. The reasons for this choice are: (i) the importance of the individual class, in our case language, might depend on the concrete application, and (ii) the distribution of the sizes of the classes might also vary in different settings.

To understand how algorithms fail, we also give a *confusion matrix* for some algorithms. Such a matrix has a row for each language in the test set and a column for each language classifier. Recall that, for example, for the English classifier we have a positive sample of English URLs and a negative sample of not-English URLs which comprises randomly selected URLs from German, Italian, French, and Spanish positive samples. So the row for the English test URLs can be read as: "How many of the English URLs in the test sets of English, French, German, Spanish, or Italian classifiers were classified as positive by these classifiers?" Similarly, the column for the English classifier can be interpreted as: "For which percentage of each of the five languages did the English classifier output Yes?" All numbers are given in percent. The values along the diagonal are exactly the recall $R = p(+|+)$. An example of such a confusion matrix can be found in Table VI which shows the "confusion" for the ccTLD algorithm.

Note that in this article all precision, recall, F-measure, and the percentages in confusion matrices are the averaged values of 10 test folds used for cross-validation. Averages are computed on the unrounded numbers and rounded afterwards. While presenting results all of P, R and $F1$ are multiplied by 100 to lie between 0 and 100 and rounded to the nearest integer.

In order to compare the performance of classifiers we performed one-tailed paired t-tests with significance level 0.025. For each test fold used for cross-validation we paired the F1-measure of two classifiers whose performance we want to compare and we calculated the differences of F1-measure values. After mapping the difference of paired F1-measure values on each test fold to a random variable, we took the average of the differences on 10 folds and calculated the standard deviation. By using these values we calculated t-values to perform a hypothesis testing to decide whether the performances of the two classifiers differ statistically significantly. When we mention that classifier X has statistically significant higher F1-measure value than classifier Y, it means that we performed paired t-tests for these classifiers.

4. RESULTS

In this section we present the performance of the classifiers for which ODP + SER dataset is used for training and testing. We begin by reporting the performance of two different baseline algorithms: (i) a simple top-level domain-based heuristics (Section 4.1), and (ii) IP-based algorithm (Section 4.2). Then we look at the performance of SVM, NB, ME, DT, RE, Markov, and ROS algorithms. We discuss the results in a per-feature set manner, that is, we first discuss the performance for the word-based features (Section 4.3), then the n-grams features (Section 4.4), and finally the custom-made features (Section 4.5). In Section 4.6 we compare our feature sets by giving insights about why they give different performances. Finally in Section 4.7 we present a comparison of the best performing classifiers along various dimensions such as the impact of training size.

4.1. Baseline: ccTLD

We present the results for the simple baseline which uses only the country code top-level domains. For this ccTLD algorithm we obtained the lowest F1-measure value (37) for English and the highest F1-measure (92) for German. Table V presents the performance summaries of this heuristic on ODP + SER test set.

Not surprisingly precision is always very high as, for example, there will not be many Italian pages in the `.fr` domain. This is also confirmed by the confusion matrix in Table VI. However, recall is very low, falling as low as 23 for English. The ccTLD+ classifier, which includes .com and .org in English ccTLDs, improves recall for English (given in parentheses), but does not affect the performance for the other languages.

Table V. Results for ccTLD Algorithm on ODP + SER Test Set

Classifier language	P	$R = p(+\|+)$	$p(-\|-)$	$F1$
English	96.6 (71.6)	22.8 (91.1)	99.2 (63.8)	**36.9** (80.2)
German	98.5	85.4	98.7	**91.5**
French	99.8	36.8	99.9	**53.7**
Spanish	99.9	37.2	99.9	**54.2**
Italian	99.8	61.3	99.9	**75.9**

Numbers in parentheses for the English classifier refer to the setting where .com and .org are counted as English TLDs.

Table VI. Confusion Matrix for the Simple ccTLD Heuristics

Language of test URLs	Reported language by binary classifiers (%) En.	Ge.	Fr.	Sp.	It.
English	23% (91%)	1%	0%	0%	0%
German	0% (10%)	85%	0%	0%	0%
French	3% (50%)	3%	37%	0%	0%
Spanish	0% (54%)	0%	0%	37%	0%
Italian	0% (31%)	1%	0%	0%	61%

The results are for ODP + SER test set. Numbers in parentheses refer to ccTLD+ algorithm, where .com and .org are also counted as English top-level domains (ccTLD+).

For German the recall (85) is higher than for the other languages. This indicates that ccTLD is a strong signal for German. Overall one can conclude that for applications where recall is important ccTLD should *not* be used as a language classifier except, possibly, for German.

The confusion matrix in Table VI explains the recall problem for ccTLD algorithm. The rows add up to less than 100% because there are domains like .net that contributed to none of the languages. The .com and .org domain contain pages of all five languages. The ccTLD+ heuristic labels all such pages as English. This greatly improves the English recall, but decreases its precision and does not change the low recall for the other languages. In the case of Spanish only 37% of the Spanish pages are in the domains marked by the heuristic as Spanish, while 54% fall into .com or .org and 9% fall into domains that we do not assign to any language. From Table VI we observed that 3% of French pages were classified as German by the German classifier and were classified as English by the English classifier. This is due to countries like Canada, Luxembourg, and Switzerland where multiple languages are spoken. As can be seen in Table I we mapped Canada (.ca) to English-speaking country, Luxembourg (.lu) and Switzerland (.ch) to German-speaking country.

4.2. Baseline: IP

The IP-based algorithm gave the lowest F1-measure (67) for Spanish and the highest (93) for German as can be seen in Figure 1. It performed statistically significantly better than our ccTLD-based algorithm for each language on URL-based language classification. As can be observed from Figure 1 the variability of the performance is minor for both ccTLD and IP algorithm.

Interesting questions arising with IP-based classifications are, for example, "Do Web servers in Spain host mostly Spanish pages?" (this question corresponds to

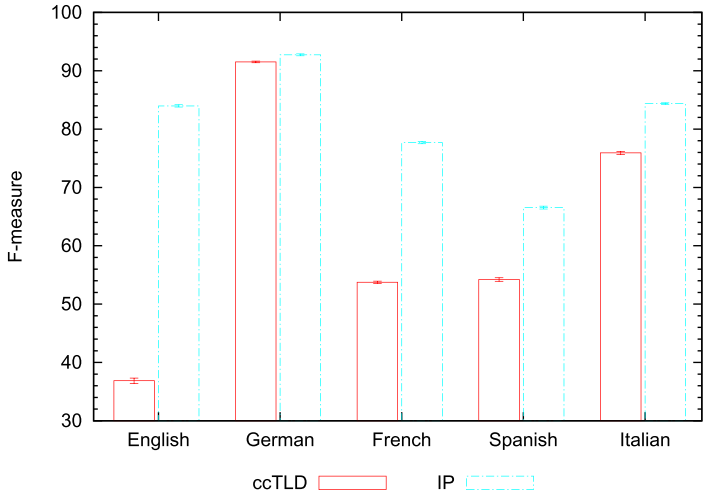

Fig. 1. F-measure values with box error bars (which extend above and below each point by one standard deviation) for ccTLD and IP algorithms.

Table VII. Results for IP Algorithm for ODP + SER Test Set

Classifier language	P	$R = p(+\|+)$	$p(-\|-)$	$F1$
English	82.1	85.9	81.3	**84.0**
German	92.8	92.7	92.8	**92.7**
French	97.5	64.6	98.4	**77.7**
Spanish	99.7	50.0	99.8	**66.6**
Italian	99.1	73.5	99.3	**84.4**

the precision of the Spanish classifier) and "Are all Spanish Web pages hosted in Spanish-speaking countries?" (this corresponds to the recall of Spanish classifier). For IP algorithm in Table VII we present the (P)recision, (R)ecall, $p(-|-)$ and F-measure values on ODP + SER test set.

From Table VII we observe that the precision values for Italian, French, and Spanish classifiers are almost close to 100. This gives an answer to the first question given before. We can say that almost all the Web pages hosted in Italy, France, and Spain are written in the languages of the corresponding countries. We cannot make the same conclusion for countries where the official language is German and English due to their lower precision values. On the other hand, for French, Spanish, and Italian classifiers we observe lower recall values than English and German. Table VII shows that half of the Spanish Web pages are not hosted in Spanish-speaking countries and it is interesting to see where they are actually hosted.

We investigate the recall problem for French, Spanish, Italian, and the precision problem for German and English in Table VIII. We present the confusion matrix for the IP classifier on ODP + SER test set. As can be seen from the first column of Table VIII, in our ODP + SER test set 37% of the Spanish Web pages, 16% of the French Web pages, and 18% of the Italian Web pages are classified as English. In other words they are hosted at servers located in English-speaking countries. We also observe that German-speaking counties host Web pages in Italian, French, and Spanish as well as German Web pages. These observations explain the low recall values for Spanish (50),

Table VIII. Confusion Matrix for IP Algorithm on ODP + SER Test Set

Language of test URLs	Reported language by binary classifiers				
	English	German	French	Spanish	Italian
English	86%	6%	1%	0%	0%
German	4%	93%	0%	0%	0%
French	**16%**	**13%**	65%	0%	1%
Spanish	**37%**	**5%**	4%	50%	1%
Italian	**18%**	**4%**	2%	0%	73%

Table IX. The Performance on ODP + SER Test Set when All the Algorithms are Used with Words as Features

Classifier language	Algo.	P	R	$p(-\|-)$	$F1$
English	NB	84.9	95.9	82.9	90.0
	ME	84.3	96.8	81.9	90.1
	RE	85.0	95.8	83.1	90.1
	ROS	73.7	63.6	77.3	68.3
	SVM	84.9	96.9	82.7	**90.5***
German	NB	97.7	94.8	97.7	96.2
	ME	97.7	94.1	97.8	95.9
	RE	97.6	95.1	97.6	**96.3***
	ROS	93.1	90.6	93.3	91.8
	SVM	98.2	94.3	98.2	96.2
French	NB	92.4	86.9	92.9	89.6
	ME	82.8	96.3	80.0	89.1
	RE	92.5	86.9	93.0	89.6
	ROS	88.8	78.2	90.1	83.1
	SVM	96.7	83.7	97.2	**89.8***
Spanish	NB	85.5	95.1	83.9	90.1
	ME	85.4	95.6	83.7	90.2
	RE	85.7	95.1	84.1	90.2
	ROS	78.5	68.5	81.2	73.2
	SVM	86.0	95.6	84.5	**90.5***
Italian	NB	95.7	87.2	96.1	91.3
	ME	95.8	86.4	96.2	90.9
	RE	95.7	87.3	96.1	**91.3***
	ROS	92.6	83.4	93.3	87.7
	SVM	97.9	85.0	98.1	91.0

French (65), and Italian (73) and explain why the precision values for English and German are lower than the other languages.

In Table VIII reported languages by binary classifiers also can be interpreted as the distribution of the languages spoken in the countries where the Web pages in our ODP + SER dataset are hosted. Furthermore, the percentage of Web pages, which are hosted in countries other than the countries we are interested in, is low.

4.3. Words as Features

We used words as features in combination with NB, ME, RE, ROS, and SVM algorithms. In Table IX we present their performance on the ODP + SER test set. For English, French, and Spanish SVM with words gave statistically significantly better F-measure values (91, 90, 91) than the other algorithms combined with words. For German and Italian classifiers RE with words performed statistically significantly

better than the other algorithms combined with words with an F1-measure of 96 and 91 respectively.

4.3.1. Impact of Domain Names. In order to understand how a classifier can use a simple feature set like words (SVM with words) to achieve an F1-measure of at least 90 for all the languages, we investigated the impact of memorizing domain names[12]. For the English classifier the percentage of test URLs whose domain was seen in the train set is 51%. This percentage is 47%, 54%, 57%, and 52% for German, French, Spanish, and Italian classifiers respectively. Based on these percentages, it is clear that domain name information definitely helps the algorithms with word features. However, this is not the only factor contributing to the strong performance for algorithms using word features because the recall of SVM words classifier, that is, for English 97, is much higher than the percentage of test URL domains seen in the train data.

In some cases domains might give contradictory or even wrong hints to the classifier. `wordpress.com` and `microsoft.com` are some examples of multilingual domains which host pages in all the languages we study. From the test set of English classifier 19% of the URLs have seen a multilingual domain in the training set. For the other languages this percentage is similar. Still for the majority of these cases, SVM with word features correctly classifies the URL.

4.4. N-Grams as Features

In our classifiers we experimented with various character n-grams in combination with NB, ME, SVM, RE, ROS, and Markov algorithms. As n-gram variants we used single n-grams like 3-, 4-, 5-, 6-grams as well as multiple n-grams like 3-4, 3-4-5, 3-4-5-6 and 3-4-5-6-7 (named as allgrams). Furthermore we applied two approaches to extract grams from URLs: (a) from words extracted from URLs, (b) directly from URLs (see Section 2.1 for details).

In Table X we present the performance of the best n-gram combination for each algorithm on ODP + SER test set. ME, SVM, RE, and ROS algorithms when combined with allgrams gave statistically significantly higher F1-measure values than the other n-gram features. As can be seen in Table X for each language, SVM with allgrams obtained from words gave statistically significantly higher F1-measure values than the other algorithms with their best n-gram combination. Thus we analyzed the performance of the SVM algorithm with different n-grams features in more detail.

In Table XI we present the F1-measure values for classifiers which use SVM as algorithm and various n-grams as feature. Classifiers which use SVM with allgrams have statistically significantly higher F-measure values than the classifiers which use SVM with other n-grams as features. Using SVM with allgrams[13] English, German, French, Spanish, and Italian classifiers have an F-measure of 94, 97, 94, 95, and 96, respectively. For comparison we also included in Table XI the F1-measure values for word features with SVM. SVM with words performed statistically significantly worse than SVM with n-gram variants for English, French, Italian, and Spanish. For German the exceptional cases for this are SVM with 3-grams-from-URL and SVM with 6-grams-from-URL.

4.5. Custom-Made Features

For representing URLs we proposed a novel feature set which consists of features extracted from URLs such as IP address, top-level domain, the count of occurrences of

[12]For example, the domain for `http://ltaa.epfl.ch/algorithms.html` is `epfl.ch` and the domain for `http://chu.cam.ac.uk/` is `cam.ac.uk`.
[13]3-4-5-6-7-grams-from-Word.

Table X. On ODP + SER Test Set the Performance of the Best N-Gram Combination for Each Algorithm

Classifier language	Algorithm	N-grams	$F1$
English	NB	allgrams-fromURL	91.9
	ME	allgrams-fromURL	93.2
	SVM	**allgrams-fromWord**	**94.2***
	RE	allgrams-fromURL	92.5
	ROS	allgrams-fromWord	79.1
	Markov	5-grams-fromURL	91.5
German	NB	6-grams-fromWord	94.9
	ME	allgrams-fromWord	96.8
	SVM	**allgrams-fromWord**	**97.2***
	RE	allgrams-fromURL	95.6
	ROS	allgrams-fromWord	89.3
	Markov	5-grams-fromURL	95.2
French	NB	6-grams-fromWord	92.7
	ME	allgrams-fromURL	93.6
	SVM	**allgrams-fromWord**	**94.4***
	RE	allgrams-fromWord	92.9
	ROS	allgrams-fromWord	82.3
	Markov	5-grams-fromURL	92.2
Spanish	NB	6-grams-fromWord	93.0
	ME	allgrams-fromURL	94.1
	SVM	**allgrams-fromWord**	**95.0***
	RE	allgrams-fromWord	93.2
	ROS	allgrams-fromWord	83.6
	Markov	5-grams-fromURL	92.5
Italian	NB	6-grams-fromWord	94.6
	ME	allgrams-fromWord	95.7
	SVM	**allgrams-fromWord**	**96.1***
	RE	allgrams-fromURL	94.9
	ROS	3-4-5-6-grams-fromWord	87.9
	Markov	5-grams-fromURL	94.3

words in dictionaries, etc. See Section 2.1 for more details about the custom-made features.

We built classifiers which use custom-made features in combination with NB, ME, RE, SVM, and DT algorithm. In Table XII we present the performance of classifiers on ODP + SER test set. DT with custom-made features achieved statistically significantly higher F-measure values than the other algorithms experimented with custom-made features. Decision Tree (DT) algorithm with custom-made features achieved an F1-measure of 94, 98, 94, 92, and 96 for English, German, French, Spanish, and Italian classifiers, respectively.

Figure 2 shows a decision tree for the English classifier. The displayed tree is a pruned version (chosen for its simplicity) only involving the top nodes of the full tree. One of the cases in which it classifies a URL as English is if the URL: (i) has an English IP (the server hosting the Web page is located in an English-speaking country), (ii) does not have any tokens in the not-English trained dictionary, and (iii) does not contain one of the Spanish ccTLDs. These conditions are intuitive as well.

Figure 3 illustrates the variability of the F1-measure values for DT with custom-made features classifier on ODP + SER dataset. For comparison we also included the

Table XI. On ODP + SER Test Set the Performance (F1-measure) of Classifiers Using SVM with Words, N-Grams and their Variants

	F1-measure of SVM classifiers				
Feature	Eng.	Ge.	Fr.	Sp.	It.
words	90.5	96.2	89.8	90.5	91.0
3-grams-fromURL	90.7	95.6	90.9	91.5	93.5
4-grams-fromURL	92.8	96.5	93.0	93.5	95.1
5-grams-fromURL	93.3	96.6	93.4	94.0	95.5
6-grams-fromURL	92.7	95.7	92.6	93.3	94.9
3-4-grams-fromURL	93.0	96.6	93.1	93.6	95.2
3-4-5-grams-fromURL	93.7	96.9	93.9	94.4	95.7
3-4-5-6-grams-fromURL	93.9	97.0	94.1	94.6	95.9
3-4-5-6-7-grams-fromURL	94.0	97.0	94.2	94.7	96.0
3-grams-fromWord	91.3	96.3	91.6	92.4	94.3
4-grams-fromWord	93.4	97.0	93.6	94.2	95.5
5-grams-fromWord	93.8	97.0	94.0	94.6	95.8
6-grams-fromWord	93.5	96.9	93.8	94.3	95.6
3-4-grams-fromWord	93.3	96.9	93.4	94.1	95.5
3-4-5-grams-fromWord	93.8	97.1	94.0	94.7	95.9
3-4-5-6-grams-fromWord	94.1	97.1	94.3	94.9	96.0
3-4-5-6-7-grams-fromWord	**94.2***	**97.2***	**94.4***	**95.0***	**96.1***

performance of SVM with words and SVM with allgrams classifiers in Figure 3. As can be observed from the plot the variability of the performance of these classifiers is minor. For all languages DT with custom-made features and SVM with allgrams achieved statistically significantly higher F1-measure values than SVM with words. German and Italian DT with custom-made features achieved statistically significantly higher F1-measure values than SVM with allgrams with values of 98 and 96, respectively. Relatively high F1-measure values of ccTLD and IP algorithms for German and Italian (see Figure 1) already indicated that ccTLD and IP features are strong signals for these languages. For French the performances of SVM with allgrams and DT with custom-made features do not statistically significantly differ with an F1-measure of 94. For English and Spanish SVM allgrams leads to higher F1-measure values than DT custom-made features classifiers with 94 and 95 points of F1-measure.

IP and ccTLD algorithms perform poorly on Spanish language. This indicates that IP and ccTLD are not strong signals for Spanish. These low signals lead to the lowest performance of DT with custom-made features on Spanish when compared to other languages. On the other hand, for German and Italian relatively high F-measure of IP and ccTLD algorithm already signaled high F-measure values of DT with custom-made features for these languages.

4.6. Comparison of Features

In this section our aim is to give a comparison of features by fixing a classification algorithm. As classification algorithm we chose SVM because with words and allgrams it gave higher F-measure values than the other algorithms. Although with custom-made features DT algorithm gave the highest performance, we used the performance of SVM algorithm for comparing features. The reasons for this choice are: (1) we did not run words and allgrams with DT algorithm as we would have obtained huge trees, (2) we wanted to compare features by fixing an algorithm and give insights about why one feature set is superior to another.

Table XIII illustrates the performance of the SVM algorithm when combined with words, allgrams, and custom-made features. For English, French, Spanish, and Italian

Table XII. The Performance of Language Classifiers on ODP + SER Test Set when NB, ME, RE, SVM and DT Algorithms are Used with Custom-Made Features

Classifier language	Algorithm	P	$R = p(+\|+)$	$p(-\|-)$	$F1$
English	NB	80.7	93.9	77.5	86.8
	ME	91.1	95.1	90.7	93.1
	RE	76.1	94.1	70.4	84.1
	SVM	90.8	95.4	90.4	93.1
	DT	92.3	95.9	92.0	**94.1***
German	NB	97.5	90.1	97.7	93.7
	ME	96.9	97.7	96.9	97.3
	RE	84.1	92.8	82.5	88.2
	SVM	97.0	97.7	96.9	97.3
	DT	97.3	98.1	97.3	**97.7***
French	NB	98.4	82.2	98.7	89.6
	ME	96.6	89.0	96.8	92.6
	RE	94.5	85.3	95.0	89.7
	SVM	96.0	89.7	96.2	92.7
	DT	96.2	92.6	96.3	**94.4***
Spanish	NB	89.7	87.7	89.9	88.7
	ME	97.8	84.3	98.1	90.5
	RE	85.6	85.2	85.7	85.4
	SVM	98.7	83.0	98.9	90.2
	DT	90.2	94.0	89.8	**92.1***
Italian	NB	99.2	87.0	99.3	92.7
	ME	98.8	93.3	98.9	96.0
	RE	94.6	88.7	95.0	91.6
	SVM	98.7	93.1	98.8	95.8
	DT	98.5	94.3	98.6	**96.4***

allgrams performed statistically significantly better than words and custom-made features. For all the languages words lead to the lower F1-measure values than allgrams and custom-made features with a minimum value for French (90) and maximum for German (96).

In order to understand why words performed worse than allgrams and custom-made features, we investigated the percentage of empty URLs. Empty URLs are test URLs that consist only of tokens which are never seen in the training set or consist of trivial tokens (“com”,“net”, “www”, “index”, “html”). We used a consistent definition of “empty URL” for all the feature sets. That is why the percentage of empty URLs presented for each language in Table XIII is identical for words, allgrams, and custom-made features. We observe the maximum percentage of empty URLs for English (19%) and minimum percentage for German (10%).

Empty URLs caused a precision problem for the English and Spanish SVM words classifier because most of the empty URLs were classified positive (See Table XIII) by these classifiers. The reason for this is the higher “com” probability in the positive language training model than the negative language training model. Empty URLs caused a recall problem for SVM words classifiers of French and Italian because almost all the empty URLs were classified negative by them. The higher “com” probability in the negative language training model than the positive language training model has an impact on the classification of empty URLs as negative by these classifiers.

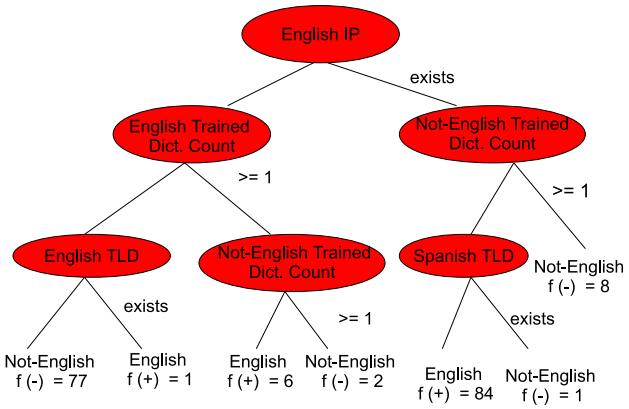

Fig. 2. A pruned version (only top nodes) of the decision tree trained on ODP + SER for English. The "English IP" refers to the feature indicating whether the URL is hosted at a Web server which is located in an English-speaking country. "Dict." refers to dictionary. At a leaf node $f(+)$ is the fraction of positive test URLs and $f(-)$ is the fraction of negative test URLs (in ODP + SER) that ended up at the leaf.

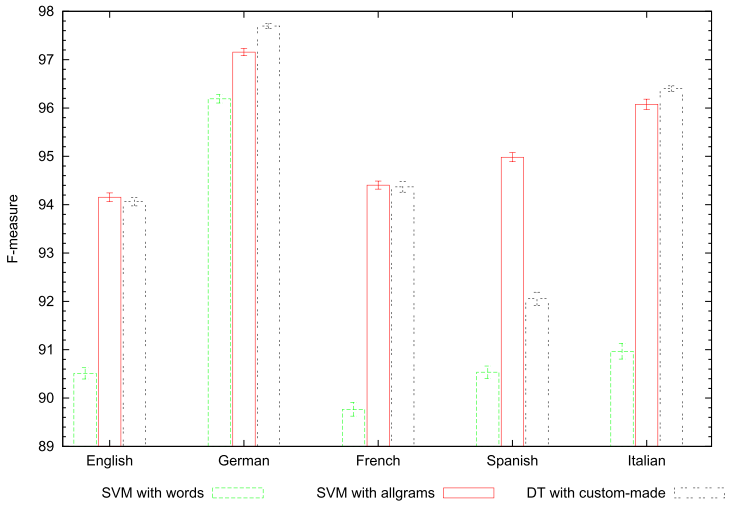

Fig. 3. F1-measure values with box error bars (which extend above and below each point by one standard deviation) for SVM with words, SVM with allgrams, and DT with custom-made features.

Allgrams and custom-made features have more information than words to cope with empty URLs. We explain this situation with example cases. For English, SVM with allgrams and SVM with custom-made features gave higher precision values than SVM with words. For example, `http://www.livredepochejeunesse.com` is an empty French test URL which only saw "com" in the training feature set. This URL was classified as English by the English SVM with words classifier because the probability of "com" is higher in the English training model than the not-English training model. On the other hand, English SVM with allgrams classifier classified this URL as not-English because it was able to detect French-specific n-grams within "livredepochejeunesse" which is a token not seen in the training feature set. Also the English SVM custom-made classifier classified it as not-English due to the fact that the server hosting this Web page is located in France. For the Spanish SVM words classifier as for the English SVM

Table XIII. When SVM is Used as an Algorithm the Performance for Words, Allgrams, and Custom-Made Features on ODP + SER

Classifier lang.	Feature	P	R	$p(-\|-)$	$F1$	% of empty URLs	% of empty and pos. classified URLs	P	R	$p(-\|-)$	$F1$
								(Empty URLs excluded)			
Eng.	Words	84.9	96.9	82.7	90.5	19.3	19.3	94.1	95.8	93.9	94.9
	Allgrams	92.2	96.2	91.9	**94.2***	19.3	14.7	95.6	96.3	95.6	**95.9***
	Custom-made	90.8	95.4	90.4	93.1	19.3	14.4	92.6	94.8	92.4	93.7
Ge.	Words	98.2	94.3	98.2	96.2	10.2	0.0	97.8	96.5	97.9	97.1
	Allgrams	98.5	95.8	98.5	97.2	10.2	0.4	98.3	97.3	98.4	**97.8***
	Custom-made	97.0	97.7	96.9	**97.3***	10.2	1.3	97.1	98.0	97.0	97.5
Fr.	Words	96.7	83.7	97.2	89.8	13.6	0.05	96.6	94.6	96.7	95.6
	Allgrams	95.8	93.0	96.0	**94.4***	13.6	4.9	97.2	96.0	97.3	**96.6***
	Custom-made	96.0	89.7	96.2	92.7	13.6	4.6	96.4	91.9	96.5	94.1
Sp.	Words	86.0	95.6	84.5	90.5	16.4	14.7	96.0	96.4	96.0	96.2
	Allgrams	95.8	94.2	95.9	**95.0***	16.4	8.8	97.6	96.9	97.7	**97.3***
	Custom-made	98.7	83.0	98.9	90.2	16.4	5.4	98.7	89.0	98.8	93.6
It.	Words	97.9	85.0	98.1	91.0	13.7	0.01	97.8	95.9	97.8	96.8
	Allgrams	97.7	94.5	97.8	**96.1***	13.7	4.5	98.4	97.4	98.4	**97.9***
	Custom-made	98.7	93.1	98.8	95.8	13.7	4.9	98.7	94.3	98.8	96.5

words we observed lower precision values than for the other features. For example, `http://www.cowboymouth.com/` is an English empty URL which is classified as Spanish by the Spanish SVM with words classifier due to higher "com" probability in the Spanish training model than the not-Spanish training model. However, the SVM allgrams classifier classified this URL as not-Spanish because it was able to detect n-grams within the "cowboymouth" token. SVM custom-made features output not-Spanish for this URL because the Web page for this URL is hosted in an English-speaking country. In summary allgrams and custom-made features contain more information than words to better classify empty URLs and they lead to higher F1-measure values.

In the article we presented our results by including empty URLs because they are part of the Web. In the real world empty URLs are common for previously uncrawled pages. Also there is a long tail of domains just like there is a long tail of queries that have never been asked before. Moreover empty URLs are more difficult URLs to classify. To make this argument more specific, consider a hypothetical word-based classifier that only knows the two tokens "tagesschau" (a German news portal) and "lemonde" (a French news portal) and removes all other tokens, giving a huge fraction of empty URLs. If empty URLs were removed from the evaluation then this classifier had both a precision and recall of 100 for French and German.

To understand whether empty URLs helped or hurt the performance of the classifiers, we also present in Table XIII the performance of the words, allgrams, and custom-made features when empty URLs are removed from the test set. The empty URLs had the biggest impact on the words feature set. When empty URLs were removed from the test sets, the precision values for the English and Spanish SVM words classifier improved roughly 9 points. The improvement in precision for these languages with the SVM words classifier is expected because most of the empty URLs were classified as positive and this led to lower precision values than the setting where empty URLs are excluded. For the French and Italian SVM with words classifier we observed an improvement in recall values when empty URLs are excluded. This is also expected because almost all empty URLs were classified as negative by these classifiers and when they are excluded the recall values improved. The exclusion of empty URLs led to 1 or 2 points improvement for allgrams and custom-made features. For German we did

not observe a big change in any of the feature sets' performance because the German words classifier already achieved a high F1-measure of 96 points and the percentage of empty URLs is lower when compared to other languages.

As can be seen from Table XIII after excluding the empty URLs from the test set, the precision is still lower for English than the other languages. This is due to English looking not-English URLs whose Web page content is written in a language other than English. This is also intuitive when considering that the dominant language on the Web is English. For example, `http://www.full-wallpaper.com/` is an English looking French URL and it is classified as English by the English SVM allgrams classifier which achieved the highest F1-measure (96) for English.

Note that in our article the training and testing data is obtained from only URLs. The challenges during URL-based language classification occur due to sparse data in URLs. A skyline approach on language classification of Web pages would be to use content of Web pages during training and testing as done in related work reported in Section 6.2. However, the scope of this work is to evaluate the success of methods that only rely on features taken from URLs.

In a similar work done by us for URL-based topic classification [Baykan et al. 2011] we also tried feature selection using information gain as a selection criterion. Information gain selects features which reveal the most information about the classes. We experimented with the features having the highest information gain values. However, none of the subset of all-grams with the highest information gain values gave an improvement in macro-averaged F-measure over using all the all-grams. Getting no improvement with this approach is not surprising as using a subset is akin to using tokens and reintroduces the problem of empty URLs. Thus we did not use the feature selection process for language-based classification.

4.7. Comparison of Various Approaches

In this study we investigate whether Web page language classification can be done only with URL. We explored this problem in various dimensions by experimenting with numerous algorithms and features. As algorithms we used machine learning algorithms (SVM, NB, ME, DT) which are widely used for text classification and similarity-based algorithms (RE, ROS, Markov) which are widely used for language identification of text. Also we applied baseline algorithms which are based on country top-level domains and IP addresses. In the previous sections we focused on the performance of each approach independently. Here we present the big picture for URL-based language classification by comparing the classifiers which achieved the highest performance among each approach along various dimensions like the impact of training size. Furthermore we present a new classifier that is a combination of the most promising baseline algorithm with the classifier that achieves the highest performance among all classifiers experimented with.

In Figure 4 we present the macro-averaged (averaged over languages) F1-measure values for the best performing classifiers and baseline classifiers. Among the machine learning algorithms Support Vector Machine (SVM) with allgrams gave the most promising results. With custom-made features, which is our novel feature set, Decision Tree (DT) gave the best performance. Among the similarity-based algorithms Relative Entropy (RE) with allgrams achieved the highest F1-measure values. On ODP + SER test set we achieved the highest macro-averaged F1-measure (95) by using SVM with allgrams and DT with custom-made features. RE with allgrams classifier gave a very similar F1-measure to SVM allgrams with a value of 94. On the other hand ccTLD- and IP-based baseline algorithms gave lower macro-averaged F1-measure values (63, 81) than SVM allgrams, DT custom-made, and RE allgrams classifiers (95, 95, 94).

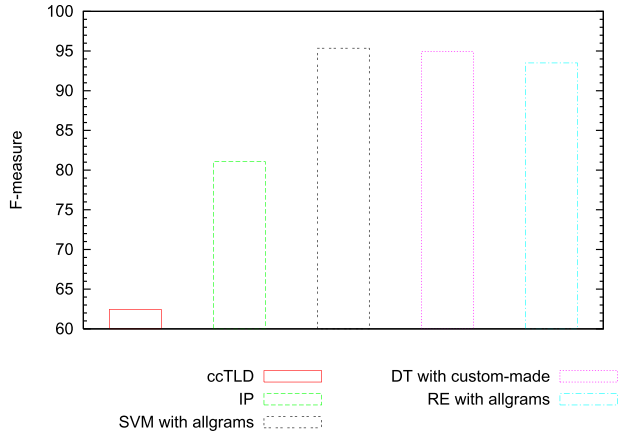

Fig. 4. Macro-averaged (averaged over languages) F1-measure values for various classifiers: ccTLD, IP, SVM with allgrams, DT with custom-made features, RE with allgrams.

4.7.1. Combining SVM with IP Algorithm. The promising macro-averaged performance (81) of the IP algorithm, that does not require any training phase, motivated to us to combine it with the best performing classifier (SVM with allgrams) to improve the performance. We name this classifier (SVM with allgrams + IP).

SVM with allgrams leads to lower recall values than precision (see Table XIV) for non-English classifiers. The question is whether we can increase recall values without sacrificing much from precision values. We combined the outputs of the SVM allgrams classifier and IP classifier by trusting IP classifier when it outputs Yes in case the SVM allgrams classifier says No. In other words, we only output No if both classifiers say No. We present the F1-measure values for SVM with allgrams + IP classifier in Table XIV. As expected, for the English classifier which does not have a recall problem, this approach led to lower F1-measure values than SVM allgrams. However for French, Spanish, and Italian this approach led to an improvement. With SVM with allgrams + IP classifier we obtained the lowest performance for English with an F1-measure of 88 and the highest performance for Italian with an F1-measure of 97. The variance of SVM with allgrams + IP classifier for each language was very small.

In Table XIV for each language, we present a performance summary for ccTLD, IP, SVM with allgrams, DT with custom-made features, RE with allgrams, and SVM with allgrams + IP classifiers. English SVM with allgrams and German DT with custom-made features performed statistically significantly better than the other mentioned classifiers in Table XIV. For French, Italian, and Spanish SVM with allgrams + IP performed statistically significantly better.

IP is the strongest feature among custom-made features and SVM with allgrams leads to the highest performance among n-gram-based classifiers. For French, Spanish, and Italian SVM with allgrams IP algorithm performs better than the other classifiers. Thus for these languages IP feature is orthogonal to allgrams.

4.7.2. Impact of Training Size. In this section we explore how the performance of the various approaches changes when the amount of training data is varied. As explained in Section 3.2 with detail, we did 10-fold cross-validation for each language on ODP + SER dataset. We trained and tested the classifiers 10 times. In each iteration different 10% of the data is used for testing and the rest of the data is used for training.

To analyze the impact of varying the training data on the performance of the classifiers, for each iteration we kept the test data fixed but randomly down-sampled the

Table XIV. Comparison of Various Approaches for Language Identification on ODP + SER Test Set

Lang.	Classifier	P	R	$p(-\|-)$	$F1$
	ccTLD	96.6	22.8	99.2	36.9
	IP	82.1	85.9	81.3	84.0
En.	**SVM with allgrams**	92.2	96.2	91.9	**94.2***
	DT with custom-made	92.3	95.9	92.0	94.1
	RE with allgrams	90.8	93.1	90.6	91.9
	SVM with allgrams + IP	80.2	98.2	75.7	88.3
	ccTLD	98.5	85.4	98.7	91.5
	IP	92.8	92.7	92.8	92.7
Ge.	SVM with allgrams	98.5	95.8	98.5	97.2
	DT with custom-made	97.3	98.1	97.3	**97.7***
	RE with allgrams	97.0	92.8	97.2	94.9
	SVM with allgrams + IP	92.7	99.0	92.2	95.7
	ccTLD	99.8	36.8	99.9	53.7
	IP	97.5	64.6	98.4	77.7
Fr.	SVM with allgrams	95.8	93.0	96.0	94.4
	DT with custom-made	96.2	92.6	96.3	94.4
	RE with allgrams	94.2	91.7	94.4	92.9
	SVM with allgrams + IP	94.6	96.6	94.5	**95.6***
	ccTLD	99.9	37.2	99.9	54.2
	IP	99.7	50.0	99.8	66.6
Sp.	SVM with allgrams	95.8	94.2	95.9	95.0
	DT with custom-made	90.2	94.0	89.8	92.1
	RE with allgrams	94.7	91.8	94.8	93.2
	SVM with allgrams + IP	95.8	96.5	95.7	**96.2***
	ccTLD	99.8	61.3	99.9	75.9
	IP	99.1	73.5	99.3	84.4
It.	SVM with allgrams	97.7	94.5	97.8	96.1
	DT with custom-made	98.5	94.3	98.6	96.4
	RE with allgrams	95.5	93.7	95.6	94.6
	SVM with allgrams + IP	97.3	98.0	97.2	**97.6***

training data with sampling ratios of 10%, 1%, and 0.1%. For each sampling ratio (training size) the reported values for the classifier performances in this section are the averaged values on 10 iterations used for cross-validation.

Figure 5 illustrates the F1-measure performance averaged over all five languages on the y-axis. The x-axis corresponds to an increase in training data, starting from 0.1% of all available training data and going up to 100%, which corresponds to a total of 450k URLs for each language. The main observations from the plot are: (1) As expected, the performance of ccTLD and IP classifiers does not change with varying the training data size because they don't involve a training phase. This is exactly what we see with the constant lines for these classifiers. (2) Decision Tree with custom-made features is more vulnerable to the reduction in training data size especially for 1% and 0.1%. (3) SVM with allgrams and RE with allgrams gives very close macro-averaged F1-measure values under all training sizes. (4) SVM with allgrams + IP gave the highest macro-averaged F1-measure (90) for the smallest amount of training data (0.1%).

5. APPLICATIONS FOR URL-BASED LANGUAGE CLASSIFICATION

When the content of a Web page is not available or it is costly to obtain it, URL-based language classifiers can be useful. In this section we apply our URL-based language

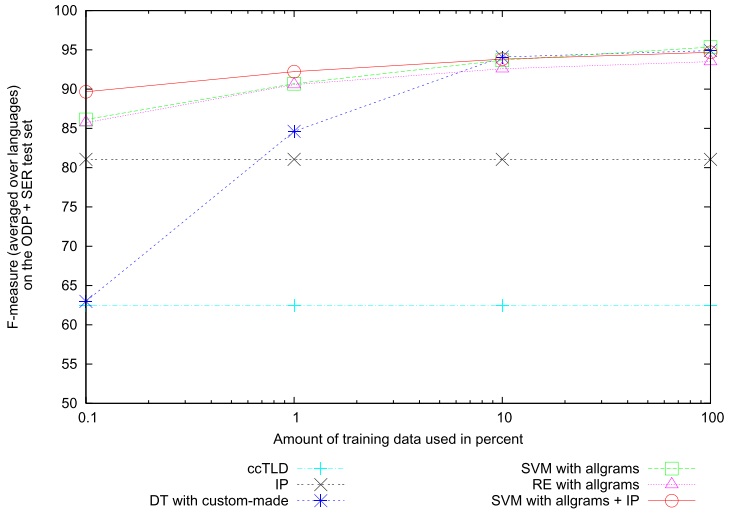

Fig. 5. A plot showing how the performance of different approaches changes on ODP + SER test set as the amount of training data is increased from a total of 450 URLs (for each language classifier) to 450k URLs.

classifiers to identify the language of Adobe Flash Web pages and the language of Web crawl pages. Recall that URLs of Flash and Web crawl pages were only used for testing purposes and they were not included in the training set. The training data obtained from ODP + SER dataset is used to train the classifiers built in this section.

5.1. Classification of Multimedia Web Pages

A useful application for the URL-based language classification is identifying the language of Web pages which have only multimedia content. For such pages it is often difficult or impossible to automatically obtain a textual representation of the content which could then be used for language classification. In this article we use Adobe Shockwave Flash Files (SWF) as an example for multimedia files. SWF files are basically graphics and animation files. Even though search engines have proprietary techniques to (attempt to) extract text from flash files to include them in their indexes, we still view this file type as representative or at least indicative of content where such extraction is infeasible.

Flash Dataset. We obtained Flash files in October 2010 for English, German, French, Spanish, and Italian by querying a commercial search engine with two different approaches. In the first approach, our queries consisted of numbers from 1 to 30 and we restricted the search engine to return URLs which have the country code Top-Level Domain (ccTLD) of the country where the required language is spoken. As ccTLDs we used `.uk` for English, `.de` for German, `.fr` for French, `.es` for Spanish, and `.it` for Italian. In the second approach we used a list of the most 10 frequently used words for each language. Table III illustrates these lists which are obtained from Wikipedia. We used 8 of these 10 words in this list in combination with numbers from 1 to 15 while cycling through the list. For both approaches we restricted the search engine to return the Web pages in the language we are interested in and to return the Flash Web pages by using the "filetype:swf" option. With the first approach we obtained around 1k URLs for each language but with the second approach we obtained less data than the first approach for non-English languages.

Table XV. Details about the Flash Dataset which is Only Used as Test Set

Data set	Language	Size
Flash	English	2,400
	German	2,200
	French	1,700
	Spanish	1,500
	Italian	1,700

In Table XV we present the total test set sizes we obtained for each language. For English we managed to get the largest amount of test size (2.4k) and for Spanish the smallest amount of test size (1.5k) among the languages. Unequal sizes of negative and positive test set sizes for a classifier do not have any impact on the performance of the classifiers due to the methodology we used for evaluation (see Section 3.3).

Note that in the experiments, the Flash dataset is not included in the training phase and it is solely used for testing purposes. In practice, this is the case for crawlers which focus on multimedia pages in specific languages. They have training data available in their classifiers and the URLs which they will encounter during crawling can be from a completely different dataset than the URLs in training set. Thus, high classification results on Flash data will be a good indicator whether our techniques for URL-based language classification can be applied to real use-cases.

In Table XVI we present the performance of ccTLD, IP, DT with custom-made, SVM with allgrams, RE with allgrams, and SVM with allgrams + IP classifiers on the Flash dataset. Recall that we used the training models obtained from ODP + SER dataset to train these classifiers. In Section 3.2 we explained in detail how the training dataset is constructed for each language. Flash URLs were only given as test URLs to the classifiers.

The best performance for each language has a performance of at least 95 points of F1-measure. For English SVM with allgrams performs equally with DT with custom-made features with an F1-measure value of 95. For German DT with custom-made features, for French and Spanish SVM with allgrams + IP classifier performs statistically significantly better than the other mentioned classifiers. For Italian DT with custom-made features and SVM with allgrams + IP classifier have the same F1-measure with a value of 98. The IP baseline algorithm achieved at least 80 points of F1-measure for all the languages. This high performance shows its highest impact when the training dataset is reduced to 0.1%. For this training size, as can be seen from Figure 6, SVM with allgrams + IP achieves 92 points of F1-measure. The performance of baseline algorithms on Flash dataset is higher than ODP + SER dataset. The performance of the classifiers on Flash dataset with the decrease in training size is similar to the performance on ODP + SER dataset. Except for the DT with custom-made features the amount of drops in the performances is smaller when compared to ODP + SER dataset.

5.2. Language-Focused Crawl

Language classifiers are useful for crawlers of Web search engines, which frequently try to satisfy certain language quotas. To determine the language of uncrawled Web pages, they have to download the page, which might be wasteful if the page is not in the desired language. With URL-based language classifiers these redundant downloads can be avoided.

In practice, one might want to train a model on one dataset, but then test it on another with very different characteristics. For example, the URLs encountered during

Table XVI. Comparison of Various Approaches for Language Identification on Flash Test Set

Lang.	Classifier	P	R	$p(-\|-)$	$F1$
	ccTLD	98.7	52.0	99.3	68.1
	IP	90.5	85.1	91.1	87.7
En.	**SVM with allgrams**	92.8	97.3	92.5	**95.0**
	DT with custom-made	93.6	96.6	93.4	**95.1**
	RE with allgrams	88.0	97.2	86.7	92.3
	SVM with allgrams + IP	87.5	99.1	85.8	92.9
	ccTLD	99.5	86.5	99.6	92.6
	IP	93.8	94.3	93.7	94.0
Ge.	SVM with allgrams	98.8	94.7	98.8	96.7
	DT with custom-made	98.1	96.4	98.1	**97.2***
	RE with allgrams	96.2	87.4	96.6	91.6
	SVM with allgrams + IP	93.2	98.1	92.9	95.6
	ccTLD	99.9	67.0	99.9	80.2
	IP	99.0	69.5	99.3	81.6
Fr.	SVM with allgrams	98.5	91.0	98.6	94.6
	DT with custom-made	95.6	92.4	95.7	94.0
	RE with allgrams	94.7	92.3	94.8	93.5
	SVM with allgrams + IP	97.9	94.4	98.0	**96.1***
	ccTLD	99.9	76.9	99.9	86.9
	IP	99.6	70.7	99.7	82.7
Sp.	SVM with allgrams	97.0	95.8	97.0	96.4
	DT with custom-made	95.1	94.7	95.1	94.9
	RE with allgrams	97.4	92.0	97.6	94.6
	SVM with allgrams + IP	96.9	97.2	96.9	**97.1***
	ccTLD	99.8	83.9	99.9	91.2
	IP	98.1	88.1	98.3	92.9
It.	SVM with allgrams	99.5	92.3	99.6	95.8
	DT with custom-made	98.4	96.8	98.4	**97.6**
	RE with allgrams	97.9	90.3	98.1	94.0
	SVM with allgrams + IP	98.1	97.0	98.1	**97.5**

a Web crawl will most likely be very different from the URLs in ODP + SER dataset, but *before* starting the crawl only the latter will be available to train a classifier. We investigated this type of scenario by training models on ODP + SER dataset and then evaluating their performance for the classification of URLs encountered during small-scale Web crawls. There are many other features one could use (analyze the content of the source page, just classify the anchor text, have majority voting using inlinks, etc.) while building a language-specific crawler. We performed these crawls as a feasibility study to evaluate the performance of our classifiers and to show that there is a potential that language-focused crawlers can benefit from our techniques.

We made three crawls for English, German, French, Spanish, and Italian. All crawls started from the same seeds but they differed in the way they selected the next Web page to crawl. In the first crawl (Random) we picked a URL uniformly at random in the queue of the uncrawled URLs. In the second crawl (ccTLD) we used the country top-level domain to select URLs from the queue. The next page to crawl was chosen uniformly at random among the pages with a matching ccTLD. In the third crawl (SVM allgrams) we used the confidence value of SVM allgrams classifier to prioritize uncrawled URLs in the queue. Note that we used the training models obtained from ODP + SER dataset to train SVM with allgrams classifiers. In Section 3.2 we explained

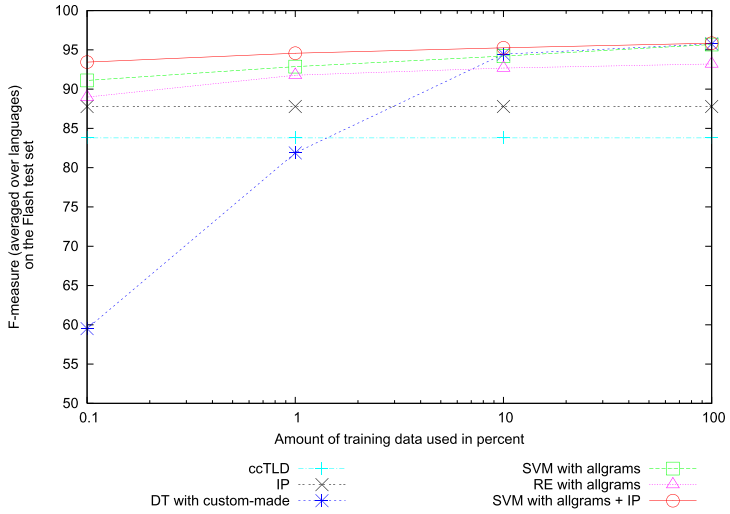

Fig. 6. A plot showing how the performance of different approaches changes on the Flash test set as the amount of training data is increased from a total of 450 URLs (for each language classifier) to 450k URLs.

Table XVII. Statistics about the Language-Specific Crawls

Lang.	Seed URL	Crawl size	# of visited hosts		
			Random	ccTLD	SVM allgrams
En.	`www.bbc.co.uk`	1,000	770	423	279
Ge.	`www.tagesschau.de`	1,000	654	470	242
Fr.	`www.lemonde.fr`	1,000	711	460	273
Sp.	`www.elmundo.es`	1,000	585	353	252
It.	`www.corriere.it`	1,000	625	474	314

in detail how the training dataset is constructed for each language. URLs encountered during the crawl were only given as test URLs to SVM allgrams classifier. All three crawls were started from five different news portals, one for each language. Table XVII illustrates the crawling seed URLs and the number of crawled Web pages.

We wanted to ensure that a large number of hosts is visited and not only pages from hosts with an "obvious" language are crawled. Thus we imposed the restriction that for each host only the first 10 discovered URLs were added to the queue and only up to 10 URLs were crawled for each host. Note that this introduces a bias against our approach. We believe that such an approach is more adequate to demonstrate the behavior of a crawler in the long run as it will jump across hosts. To define a host, we used everything between the initial "http://" and the (optional) first "/". Any "www." at the beginning was removed so that, for example, `www.twitter.com` and `twitter.com` were counted as the same host. However, `wetter.tagesschau.de` and `blog.tagesschau.de` were considered distinct hosts. In Table XVII we present the number of visited hosts in each crawl. The random crawling strategy visited more hosts than ccTLD and SVM allgrams crawler. The crawler using SVM allgrams visited the smallest number of distinct hosts. This is caused by the fact that other pages from the host pertaining to the highest confidence language classification are also likely to correspond to a high confidence and hence are likely to be crawled next. For ccTLD the next page (and host)

Table XVIII. The Percentage of Crawled Pages for which We Could not Get a Classification Label

Language	Random	ccTLD	SVM allgrams
English	4%	5%	6%
German	6%	5%	5%
French	4%	5%	8%
Spanish	6%	13%	10%
Italian	2%	2%	3%

We ignored these undownloaded Web pages while calculating precision.

Table XIX. The Percentage of Crawled Pages Written in the Desired Language (Precision)

Language	Random	ccTLD	SVM allgrams
English	94.5	95.6	95.4
German	38.4	88.7	95.4
French	49.5	92.3	96.8
Spanish	25.2	92.2	94.8
Italian	38.2	87.1	90.1

to visit is chosen uniformly at random from all the pages in the queue satisfying the ccTLD requirement.

When identifying outgoing links on a page we ignored multimedia files and style sheets and these files did not contribute to the count of 10 URLs per host. All three crawling variants were stopped after visiting 1000 distinct URLs. The crawler was implemented in Perl using standard libraries (LWP::UserAgent, HTML::LinkExtractor, URI::URL) and only URLs with the http protocol were considered.

To verify if the pages crawled did indeed belong to the target language, in other words to identify the languages of the pages from content, we used the Lingua::LanguageGuesser Perl module that is developed in Chan and Yamana [2010]. At least the differences in precision between ccTLD and SVM allgrams crawlers might be so small and well within the uncertainty/error bounds of the classifier. We could not get a classification for a small percentage of pages mentioned in Table XVIII because we could not download their content for evaluation with Lingua::LanguageGuesser.

In Table XIX we present the percentage of crawled pages written in the desired language, in other words, we report the precision for the crawlers. There is strong evidence that language-focused crawlers could benefit from our techniques because for each language with SVM allgrams crawler we obtained at least 90 points of precision. We obtained the lowest precision for Italian (90) and the highest precision (97) for French. For each language Random crawler achieved lower precision values than ccTLD and SVM allgrams crawler. This result is as expected as we did not put a language restriction during Random crawling.

As can be seen from Table XIX the precision for ccTLD crawler is very high. However, the low recall for ccTLD baseline classifier from earlier experiments in Section 4.1 shows that for the classifier for language X there is a high percentage of URLs that belong to the language X but not to the TLD that ccTLD would classify as language X. In other words ccTLD crawler would never find 77% of the English pages in the test set of the English classifier which do not have an English ccTLD. Similarly it will not be able to crawl 63% of the French pages, 63% of the Spanish pages, 39% of Italian pages, and 15% of German pages in the test sets of the respective classifiers.

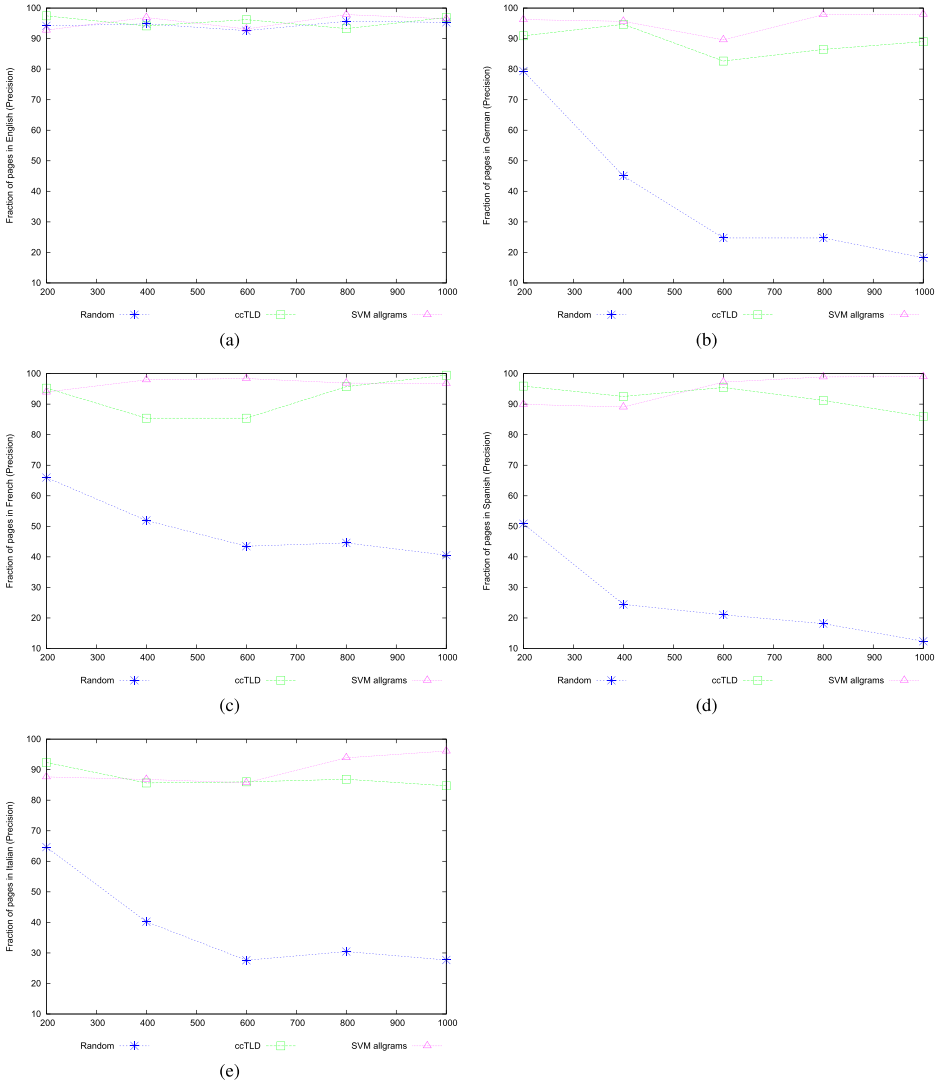

Fig. 7. Plot of the fraction of pages crawled in the desired language (precision) "over time". We computed the fraction for the first 200, then the next 200, and so on. (a) English; (b) German; (c) French; (d) Spanish; (e) Italian.

In Figure 7 we present the plot for the fraction of pages crawled in the desired language "over time". We computed the fraction for the first 200, then the next 200, and so on. For German, Italian, French, and Spanish the fraction of the Random crawler got lower as the crawl moved away from the original seed. On the other hand, for SVM allgrams and ccTLD crawler the fraction is pretty stable.

The most interesting observation from Figure 7 is the fact that English Random crawler has much higher precision than Random crawlers for the other languages. In each time interval the precision for English Random crawler is at least 90 points. One explanation for the high degree of English pages in the Random crawl is that English pages hardly ever link to non-English pages. On the other hand, non-English pages link to English pages more frequently. Supporting this hypothesis, there is a study

on 4,000 Web sites by Halavais [2000] to mine the links between sites in different languages and countries. This study determined that the percentage of Web pages linking to target pages in a different language than the source page is lower for English source pages than non-English source pages.

6. RELATED WORK

Language identification is a well-studied problem. An excellent survey of language identification can be found in Sibun and Reynar [1996] and Hughes et al. [2006]. The related work about language classification, content-based Web page language classification, and URL-based Web page language classification are reviewed in Sections 6.1, 6.2, and 6.3 respectively. These sections are grouped by the features they use for language classification. Finally in Section 6.4 URL-based other related work is presented.

6.1. Generic Language Classification

When there is a sufficiently long sample of text, usage of common short words as language indicators [Ingle 1976], for example, the word "the" indicates English, works well. Variants of this approach look for discriminative letter sequences, n-grams, such as "ery" for English or "eux" for French. However, neither of these ideas are directly applicable to our setting of URL language classification, where the text is very short and does not even have to contain any proper words.

Rank Order Statistics [Cavnar and Trenkle 1994], Relative Entropy [Sibun and Reynar 1996], and Markov [Dunning 1994] algorithms use n-grams as features for language classification. These algorithms first form language models from the training data, then classify a test document with the label of the closest language model to the test document. They use different metrics for distance. The Rank Order Statistics algorithm uses the rank difference between the most frequent n-grams common in the language model and the test document as distance. The Relative Entropy algorithm uses relative entropy between the probability distribution of the language model and the test document as distance measure. The Markov algorithm uses character-based Markov models for language identification. A test document is labeled with the language which maximizes the probability of generating n-grams in the test document from its previous n-1 grams. See Section 2.2 for more details about these algorithms. Markov models are also used to train Prediction by Partial Match compression models in Teahan and Harper [2001], where they classify a document based on compression performance. On newsgroup articles in the Usenet dataset Cavnar and Trenkle [1994] obtained an F1-measure of 99 points by using a combination of 1 to 5 grams. In Dunning [1994] for Markov and in Sibun and Reynar [1996] using bigrams for Relative Entropy 99.9 points of accuracies are reported. However, there is no consensus about the performances of Rank Order Statistics [Cavnar and Trenkle 1994], Relative Entropy [Sibun and Reynar 1996], and Markov [Dunning 1994] algorithms on the same dataset and the performance on short texts is not clear. We used these algorithms in our experiments to evaluate their performance on URL-based language classification.

Grefenstette [1995] compared the performance of trigrams and word features with an algorithm similar to Naïve Bayes. It was observed that trigram-based techniques outperform the word-based techniques when the text is short and perform no worse on longer texts. Based on the idea of common/stop-words for languages, authors in Rhekurek and Kolkus [2009] proposed a method which determines relevancy scores for words in a language indicating how specific the words are for the language. They showed that their method gives competitive results with n-grams on identifying European languages on a small dataset.

6.2. Content-Based Web Page Language Classification

Studies in Martins and Silva [2005] and Vega and Bressan [2001] use n-grams for identifying the language of Web pages using their content. To index Portuguese Web pages in a search engine Martins and Silva [2005] built a language classifier by using a combination of n-grams from 1 to 5 with a Rank Order Statistics algorithm [Cavnar and Trenkle 1994] and by using heuristics for better handling Web data. If a Web page contained a language tag they took it into consideration if it indicates a language other than English. In their experiments they ignored and did not label Web pages having less than 40 characters length. They obtained a recall of 95 and (mapped to our evaluation setup) a precision of 99. This gives an F1-measure of 97, comparable to the F1-measure of our Italian and German classifiers (see Section 4.7). Vega and Bressan [2001] built an Indonesian classification system for use in a search engine for the Indonesian Web. They trained a trigram-based classifier on a dictionary of 10k Indonesian words. On a small test set of 24 documents they achieved a precision and recall of around 90.

Studies in Pingali et al. [2006] and Somboonviwat et al. [2005] used the character encoding of Web pages and link information for building language-specific crawlers. Authors in Pingali et al. [2006] used the content of Web pages, character encoding, and link information like Chakrabarti et al. [1999] to build a language-specific crawler for Indian Web pages which have different encoding than ASCII. Somboonviwat et al. [2005] built a language-specific Web crawler for Japanese and Thai. Their crawler used the character encoding scheme of Web pages together with the frequency of certain bytes for language classification. They simply assume that this works perfectly to detect the language when: (i) the full text is available and (ii) the languages under consideration use a complex character encoding. Unfortunately, they did not state how well the language identification part of their crawler worked. Their crawling strategies are based on the observation that Web pages written in the same languages tend to be close to each other in the hyperlink structure of the Web. This fact was exploited in Hayati [2004] to improve the classification for European languages. Note that we did not use the link structure of the Web in this article as constructing it requires retrieving the content of Web pages.

A recent work [Baldwin and Lui 2010] used Support Vector Machine (SVM) and Naïve Bayes algorithms with n-grams to classify 17 European languages from Wikipedia. The authors demonstrated that SVM gives very good performance especially on short text and showed that language classification is much more difficult for shorter documents.

6.3. URL-Based Web Page Language Classification

Tamura et al. [2007] and Chan and Yamana [2010] present techniques for building language-specific crawlers by using the URLs of the hyperlinks extracted from the crawled pages. As features, hostnames or TLDs extracted from URLs are used. Tamura et al. [2007] proposed a method for language-specific crawling for Thai language. The hostnames of the URLs extracted from the outlinks of crawled Web pages are used to decide whether to put these links into crawling queue or not. The new discovered links are discarded if they belong to a host nonrelevant to the language. The relevancy scores for hosts are calculated during the crawl after determining whether a crawled page is indeed in the desired language. The authors obtained 70 points of precision with this method Thai language. Authors in Chan and Yamana [2010] built URL-based language-specific crawlers for Japanese, Chinese, and Korean. Their crawler adds the new discovered links to the crawling queue if the TLD of the URL belongs to the desired language. Otherwise the crawler adds the URL

to the queue if the language of the anchor text of the hyperlink is the desired language for the crawl. This crawling strategy leads to 10 points of improvement in precision for Korean language when compared to breadth-first crawling which achieved 61 points of precision. They only obtained an improvement of 1 point of precision for Chinese and Japanese as they linked to nonlanguage pages less frequently than Korean.

Custom-made features for URL-based language classification are used at Baykan et al. [2008] and Alabbad and Aounallah [2010]. In our previous work on URL-based language classification [Baykan et al. 2008] we used the custom-made features mentioned in Section 4.5 except for IP feature. Note that this study extends Baykan et al. [2008] with a more comprehensive evaluation setup (10-fold cross-validation, statistical significance tests), with additional algorithms (SVM, IP, combination of IP and SVM), with additional features (various single gram sizes, combination of multiple n-grams, inclusion of IP in custom-made features), and with the evaluation of applications like language-focused crawlers and multimedia files. The authors of Alabbad and Aounallah [2010] built binary classifiers with machine learning algorithms using custom-made features to identify Arabic Web pages. For example, some of these custom-made features indicate whether a URL contains "arab", contains "_ar", and contains digits frequently used instead of Arabic letters in URLs. They achieved a recall of 31 and a precision of 84 with the Naïve Bayes algorithm on a small dataset. The challenge for Arabic language is the fact that it does not use the Latin alphabet, which is used for URLs.

6.4. URL-Based Other Related Work

There are papers on classifying Web pages according to topics using only URLs. In Kan [2004] the author tries to classify Web pages from academic hosts according to the categories "course", "faculty", "project", or "student". Although their study is similar in spirit (uses only the URL), the actual problem (classify according to language versus category) and the datasets (1.25M versus 5k pages from 4 universities) and algorithms differ considerably. In Hänse et al. [2010] a classifier based on only URLs is developed to determine whether a Web site is about a scientific conference or not. They achieved an accuracy of 96 points by using Maximum Entropy algorithm in combination with features extracted from URLs like URL length, words in URL and their lengths. In Baykan et al. [2009b, 2011] authors developed methods to classify Web pages by using URLs into topics like Sports, Arts, News, Shopping, etc. They showed that the inherent overlap between topics and sparse information in URLs makes URL-based topic classification a very challenging problem.

Chung et al. [2010] present techniques for classifying hostnames of Web pages according to spam topics using only URLs. They experimented on a Japanese dataset by using machine learning algorithms in combination with n-grams and words. To obtain true labels for hostnames the authors assumed that hostnames of Web pages in the same strongly connected component have the same topic. Kumar and Tomkins [2010] propose a new taxonomy of pageviews. As stated by the authors using URLs of Web pages for automatically classifying pages into nodes of taxonomy would be useful for obtaining more classified pages than doing the classification manually.

Authors in Anastácio et al. [2009] presented an approach by using n-grams extracted from URLs to classify documents according to locational relevance. Freudiger et al. [2009] use URLs of Web pages to decide whether they contain potentially sensitive information from which online advertisers can benefit by putting third-party cookies during browsing. Koppula et al. [2010] present techniques to find rules from URLs to identify duplicate Web pages. In Umbrich et al. [2009] techniques for identifying multimedia files only from URL are proposed. These techniques were applied in a focused crawler for media-type targeted search engines.

7. CONCLUSIONS

We applied a variety of algorithms and feature sets to a real dataset obtained from the Open Directory Project and the search results of a commercial search engine (ODP + SER dataset). In addition to applying state-of-the-art algorithms and features for language and text classification we also proposed a novel feature set (custom-made features) and novel algorithms (IP, SVM with allgrams + IP). We compared the performances of the different URL-based language classifiers along various dimensions such as features, algorithms, and training size. Additionally we tested our best performing classifiers on ODP + SER dataset on the classification of multimedia Web pages and in small-scale language-focused crawlers. Here we summarize our main results for URL-based Web page language classification.

Baseline algorithms that use country code top-level domains (ccTLD algorithm) and that use the IP address extracted for the host name of a URL (IP algorithm) cannot be used in applications where recall is important. Although these algorithms have the nice property of not requiring any training phase they generally lead to low recall and high precision values. The low recall values for the ccTLD algorithm are explained by the fact that a high percentage of Web pages [Baykan et al. 2009a] are in ".com" or ".org" or ".net". These three domains do not belong to the set of language specific ccTLDs. Although the IP algorithm leads to more promising results than ccTLD algorithm it fails to classify French, Spanish, and Italian Web pages which are hosted in English- or German-speaking countries. This fact explains the low recall values of Italian, French, and Spanish and also the relatively low precision values of English and German. High F1-measure values for ccTLD and IP algorithm for German show that ccTLD and IP are strong signals for this language.

To build URL-based language classifiers we experimented with various combinations of algorithms and features. Our statistically significant results can be summarized as follows. (1) Among all the algorithms combined with features sets based on words or on variants of n-grams the SVM algorithm gave the most promising performance. Particularly when combined with allgrams, SVM achieved the highest F1-measure values, by having a minimum value of 94 for English and a maximum value of 97 for German on ODP + SER dataset. (2) Combining SVM with IP the F1-measure improved by 1–2 points for all languages except for English and German. Even on multimedia data, that is, the Flash test dataset, which is only used for testing purposes and *not* for training, we obtained F1-measures of at least 95 points with the aforementioned classifiers. (3) For custom-made features, which is our novel feature set that consists of information such as IP-addresses, ccTLD, number of words in various dictionaries (city, thesaurus, etc.), the decision tree algorithm led to the highest F1-measure values among all the algorithms with this feature set. German DT with custom-made features achieved even higher F1-measure values than SVM with allgrams both on the ODP + SER dataset (98) and on the Flash test set (97). (4) The results for SVM with allgrams and its variant (SVM with allgrams + IP) are quite stable as the size of the training data is reduced giving hope that it would work well if applied to the Web in general. On the other hand, DT with custom-made features is more vulnerable to the reduction in training data size.

Additionally we made the following interesting observations. We compared words, allgrams, and custom-made features by experimenting with a fixed algorithm (SVM). Words features lead to lower F1-measure values than allgrams and custom-made features. To better understand this behavior we investigated how different features classified *empty URLs*. Empty URLs are test URLs that consist only of tokens which are never seen in the training set or consist of trivial tokens ("com","net", "www", "index", "html"). To classify empty URLs algorithms with a word-based feature set can only

use statistics of trivial tokens, such as “com”, during classification. On the other hand, algorithms using the allgrams feature set applied language-specific n-grams within tokens that it has not seen in the training data. Algorithms with the custom-made feature set used the IP address of the URL to detect the language spoken in the country where the Web page is hosted. In other words allgrams and custom-made features have more information than words to cope with empty URLs.

We applied our best performing URL-based language classifiers to identify the language of Adobe Flash Web pages and the language of Web crawl pages. These test sets were only used for testing, and *not* included in the training phase of the classifiers. On the classification of Flash pages we achieved the lowest F1-measure for English (95) by using SVM with allgrams classifier and the highest F1-measure for Italian (98) by using SVM with allgrams + IP classifier. On small-scale language-focused crawls we obtained the lowest precision for Italian (90) and the highest precision (97) for French by using SVM allgrams classifier. High classification results on these real use-cases show that there is a high potential that multimedia Web page classifiers and language-specific crawlers can benefit from our best performing URL-based language classifiers.

REFERENCES

Alabbad, S. and Aounallah, M. 2010. URL based classification of arabic web pages. In *Proceedings of the International Conference on Internet Computing (ICOMP).* 334–338.

Anastacio, I., Martins, B., and Calado, P. 2009. Classifying documents according to locational relevance. In *Proceedings of the Portuguese Conference on Artificial Intelligence (EPIA).* 598–609.

Baldwin, T. and Lui, M. 2010. Language identification: The long and the short of the matter. In *Proceedings of the Human Language Technologies: Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT).* 229–237.

Baykan, E., Castelberg, S. D., Henzinger, M., Keller, S., and Kinzler, M. 2006. A comparison of techniques for sampling web pages. In *Proceedings of the International Workshop on Information Integration on the Web (IIWEB).*

Baykan, E., Henzinger, M., and Weber, I. 2008. Web page language identification based on URLs. *Proc. VLDB Endow. 1,* 176–187.

Baykan, E., Henzinger, M., Keller, S., Castelberg, S. D., and Kinzler, M. 2009a. A comparison of techniques for sampling web pages. In *Proceedings of the International Symposium on Theoretical Aspects of Computer Science (STACS).* 13–30.

Baykan, E., Henzinger, M., Marian, L., and Weber, I. 2009b. Purely URL-based topic classification. In *Proceedings of the International Conference on World Wide Web (WWW)* (Poster Track). 1109–1110.

Baykan, E., Henzinger, M., Marian, L., and Weber, I. 2011. A comprehensive study of features and algorithms for URL-based topic classification. *Trans. Web 5*, 3, 15:1–15:29.

Cambazoglu, B. B., Varol, E., Kayaaslan, E., Aykanat, C., and Baeza-Yates, R. 2010. Query forwarding in geographically distributed search engines. In *Proceedings of the International Conference on Research and Development in Information Retrieval (SIGIR).* 90–97.

Cavnar, W. B. and Trenkle, J. M. 1994. N-Gram-Based text categorization. In *Proceedings of the Symposium on Document Analysis and Information Retrieval (SDAIR).* 161–175.

Chakrabarti, S., Van Den Berg, M., and Dom, B. 1999. Focused crawling: A new approach to topic-specific web resource discovery. *Comput. Netw. 31*, 11, 1623–1640.

Chan, S. -B. and Yamana, H. 2010. The method of improving the specific language focused crawler. In *Proceedings of the CIPS-SIGHAN Joint Conference on Chinese Language Processing (CLP).*

Chung, Y., Toyoda, M., and Kitsugeregawa, M. 2010. Topic classification of spam host based on urls. In *Proceedings of the Forum on Data Engineering and Information Management (DEIM).*

Dunning, T. 1994. Statistical identification of language. Tech. rep., Computing Research Lab (CRL), New Mexico State University.

Freudiger, J., Vratonjic, N., and Hubaux, J. -P. 2009. Towards privacy-friendly online advertising. In *Proceedings of the Web 2.0 Security and Privacy Conference (W2SP).*

Grefenstette, G. 1995. Comparing two language identification schemes. In *Proceedings of the International Conference on Statistical Analysis of Textual Data (JADT).* 263–268.

Halavais, A. 2000. Halavais, A. 2000. National borders on the world wide web. *New Media Soc. 2*, 7–28.

Hanse, M., Kan, M. -Y., and Karduck, A. 2010. Kairos: Proactive harvesting of research paper metadata from scientific conference web sites. In *Proceedings of the International Conference on Asia-Pacific Digital Libraries (ICADL)*. 226–235.

Hastie, T., Tibshirani, R., and Friedman, J. H. 2001. *The Elements of Statistical Learning*. Springer.

Hayati, K. 2004. Language identification on the world wide web. Master's project, University of California, Santa Cruz.

Hughes, B., Baldwin, T., and Bird, S. 2006. Reconsidering language identification for written language resources. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. 485–488.

Ingle, N. C. 1976. A language identification table. *Incorporated Linguist 15*, 4, 98–101.

Joachims, T. 2009. SVM-perf: Support vector machine for multivariate performance measures. http://svmlight.joachims.org/svm_perf.html

Kan, M. -Y. 2004. Web page classification without the web page. In *Proceedings of the International World Wide Web Conference on Alternate Track Papers and Posters (WWW Alt.)*. 262–263.

Koppula, H. S., Leela, K., Agarwal, A., Chitrapura, K. P., Garg, S., and Sasturkar, A. 2010. Learning url patterns for webpage de-duplication. In *Proceedings of the International Conference on Web Search and Data Mining (WSDM)*. 381–390.

Kumar, R. and Tomkins, A. 2010. A characterization of online browsing behavior. In *Proceedings of the International Conference on World Wide Web (WWW)*. 561–570.

Martins, B. and Silva, M. J. 2005. Language identification in web pages. In *Proceedings of the Symposium on Applied Computing (SAC)*. 764–768.

Math Works. 2013. Matlab. http://www.mathworks.com/products/matlab/.

McCallum, A. K. 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow

Nigam, K., Lafferty, J., and MacCallum, A. 1999. Using maximum entropy for text classification. In *Proceedings of the International Joint Conference on Artificial Intelligence Workshop on Machine Learning for Information Filtering (IJCAI)*. 61–67.

Pingali, P., Jagarlamudi, J., and Varma, V. 2006. Webkhoj: Indian language IR from multiple character encodings. In *Proceedings of the International Conference on World Wide Web (WWW)*. 801–809.

Poola, K. L. and Ramanujapuram, A. 2007. Techniques for keyword extraction from URLs using statistical analysis. US patent application. http://www.faqs.org/patents/app/20090089278.

Rhekurek, R. and Kolkus, M. 2009. Language identification on the web: Extending the dictionary method. In *Proceedings of the International Conference on Computational Linguistics and Intelligent Text Processing (CICLing)*. 357–368.

Sebastiani, F. 2002. Machine learning in automated text categorization. *ACM Comput. Surv. 34*, 1–47.

Sibun, P. and Reynar, J. C. 1996. Language identification: Examining the issues. In *Proceedings of the Symposium on Document Analysis and Information Retrieval (SDAIR)*. 125–135.

Somboonviwat, K., Kitsuregawa, M., and Tamura, T. 2005. Simulation study of language specific web crawling. In *Proceedings of the International Conference on Data Engineering Workshops (ICDEW)*. 1254.

Tamura, T., Somboonviwat, K., and Kitsuregawa, M. 2007. A method for language-specific web crawling and its evaluation. *Syst. Comput. Japan 38*, 10–20.

Teahan, W. and Harper, D. 2001. Using compression-based language models for text categorization. In *Proceedings of the Workshop on Language Modeling and Information Retrieval*.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. 2005. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res. 6*, 1453–1484.

Umbrich, J., Karnstedt, M., and Harth, A. 2009. Fast and scalable pattern mining for media-type focused crawling. In *Proceedings of the Knowledge Discovery, Data Mining, and Machine Learning Workshop (KDML)*. 119–126.

Vega, V. B. and Bressan, S. 2001. Continuous-Learning weighted trigram approach for indonesian language distinction: A preliminary study. In *Proceedings of the International Conference on Computer Processing of Oriental Languages (ICCPOL)*.

Received September 2011; revised July 2012; accepted October 2012

View publication stats