Journal of Artificial Intelligence Research 65 (2019) 675–782 Submitted 10/2018; published 08/2019

Automatic Language Identification in Texts: A Survey

Tommi Jauhiainen TOMMI.JAUHIAINEN@HELSINKI.FI
Department of Digital Humanities
The University of Helsinki

Marco Lui SAFFSD@GMAIL.COM
School of Computing and Information Systems
The University of Melbourne

Marcos Zampieri MARCOS.ZAMPIERI@RIT.EDU
College of Liberal Arts
Rochester Institute of Technology

Timothy Baldwin TB@LDWIN.NET
School of Computing and Information Systems
The University of Melbourne

Krister Lindén KRISTER.LINDEN@HELSINKI.FI
Department of Digital Humanities
The University of Helsinki

Abstract

Language identification ("LI") is the problem of determining the natural language that a document or part thereof is written in. Automatic LI has been extensively researched for over fifty years. Today, LI is a key part of many text processing pipelines, as text processing techniques generally assume that the language of the input text is known. Research in this area has recently been especially active. This article provides a brief history of LI research, and an extensive survey of the features and methods used in the LI literature. We describe the features and methods using a unified notation, to make the relationships between methods clearer. We discuss evaluation methods, applications of LI, as well as *off-the-shelf* LI systems that do not require training by the end user. Finally, we identify open issues, survey the work to date on each issue, and propose future directions for research in LI.

1. Introduction

Language identification ("LI") is the task of determining the natural language that a document or part thereof is written in.[1] Recognizing text in a specific language comes naturally to a human reader familiar with the language. Table 1 presents excerpts from Wikipedia articles in different languages on the topic of Natural Language Processing ("NLP"), labeled

1. This survey concentrates solely on the language identification of digital text, and all references to "language identification" in this survey refer to text, unless otherwise explicitly stated. For a recent overview of spoken language identification, see (Li, Ma, & Lee, 2013).

English	Natural language processing is a field of computer science, artificial intelligence, and linguistics concerned with the interactions between computers and human (natural) languages.
Italian	L'Elaborazione del linguaggio naturale è il processo di trattamento automatico mediante un calcolatore elettronico delle informazioni scritte o parlate nel linguaggio umano o naturale.
Chinese	自然語言處理是人工智慧和語言學領域的分支學科。
Japanese	自然言語処理は、人間が日常的に使っている自然言語をコンピュータに処理させる一連の技術であり、人工知能と言語学の一分野である。

Table 1: Excerpts from Wikipedia articles on NLP in different languages.

according to the language they are written in. Without referring to the labels, readers of this article will certainly have recognized at least one language in Table 1, and many are likely to be able to identify all the languages therein.

Research into LI aims to mimic this human ability to recognize specific languages. Over the years, a number of computational approaches have been developed that, through the use of specially-designed algorithms and indexing structures, are able to infer the language being used without the need for human intervention. The capability of such systems could be described as super-human: an average person may be able to identify a handful of languages, and a trained linguist or translator may be familiar with many dozens, but most of us will have, at some point, encountered written texts in languages they cannot place. However, LI research aims to develop systems that are able to identify *any* human language, a set which numbers in the thousands (Simons & Fennig, 2017).

In a broad sense, LI applies to any modality of language, including speech, sign language, and handwritten text, and is relevant for all means of information storage that involve language, digital or otherwise. However, in this survey we limit the scope of our discussion to LI of written text stored in a digitally-encoded form.

Research to date on LI has traditionally focused on *monolingual* documents (Hughes, Baldwin, Bird, Nicholson, & MacKinlay, 2006) (we discuss LI for multilingual documents in Section 10.6). In monolingual LI, the task is to assign each document a unique language label. Some work has reported near-perfect accuracy for LI of large documents in a small number of languages, prompting some researchers to label it a "solved task" (McNamee, 2005). However, in order to attain such accuracy, simplifying assumptions have to be made, such as the aforementioned monolinguality of each document, as well as assumptions about the type and quantity of data, and the number of languages considered.

The ability to accurately detect the language that a document is written in is an enabling technology that increases accessibility of data and has a wide variety of applications. For example, presenting information in a user's native language has been found to be a critical factor in attracting website visitors (Kralisch & Mandl, 2006). Text processing techniques developed in natural language processing and Information Retrieval ("IR") generally presuppose that the language of the input text is known, and many techniques assume that all documents are in the same language. In order to apply text processing techniques to

real-world data, automatic LI is used to ensure that only documents in relevant languages are subjected to further processing. In information storage and retrieval, it is common to index documents in a multilingual collection by the language they are written in, and LI is necessary for document collections where the languages of documents are not known a-priori, such as for data crawled from the World Wide Web. Another application of LI that predates computational methods is the detection of the language of a document for routing to a suitable translator. This application has become even more prominent due to the advent of Machine Translation ("MT") methods: in order for MT to be applied to translate a document to a target language, it is generally necessary to determine the source language of the document, and this is the task of LI. LI also plays a part in providing support for the documentation and use of low-resource languages. One area where LI is frequently used in this regard is in linguistic corpus creation, where LI is used to process targeted web crawls to collect text resources for low-resource languages.

A large part of the motivation for this article is the observation that LI lacks a "home discipline", and as such, the literature is fragmented across a number of fields, including NLP, IR, machine learning, data mining, social media analysis, computer science education, and systems science. This has hampered the field, in that there have been many instances of research being carried out with only partial knowledge of other work on the topic, and the myriad of published systems and datasets.

Finally, it should be noted that this survey does not make a distinction between languages, language varieties, and dialects. Whatever demarcation is made between languages, varieties and dialects, a LI system is trained to identify the associated document classes. Of course, the more similar two classes are, the more challenging it is for a LI system to discriminate between them. Training a system to discriminate between similar languages such as Croatian and Serbian (Ljubešić & Kranjcić, 2014), language varieties like Brazilian and European Portuguese (Zampieri & Gebre, 2012), or a set of Arabic dialects (Zampieri, Tan, Ljubešić, Tiedemann, & Nakov, 2015) is more challenging than training systems to discriminate between, for example, Japanese and Finnish. Even so, as evidenced in this article, from a computational perspective, the algorithms and features used to discriminate between languages, language varieties, and dialects are identical.

In this survey, we will examine the common themes and ideas that underpin research in LI. We begin with a brief history of research that has led to modern LI (Section 3), and then proceed to review the literature, first introducing the mathematical notation used in the article (Section 4), and then providing synthesis and analysis of existing research, focusing specifically on the representation of text (Section 5) and the learning algorithms used (Section 6). We examine the methods for evaluating the quality of the systems (Section 7) as well as the areas where LI has been applied (Section 8), and then provide an overview of "off-the-shelf" LI systems (Section 9). We conclude the survey with a discussion of the open issues in LI (Section 10), enumerating issues and existing efforts to address them, as well as charting the main directions where further research in LI is required.

2. LI as Text Categorization

LI is in some ways a special case of text categorization, and previous research has examined applying standard text categorization methods to LI (Cavnar & Trenkle, 1994; Elworthy, 1998).

Sebastiani (2002, Section 2.1) provides a definition of text categorization, which can be summarized as the task of mapping a document onto a pre-determined set of classes. This is a very broad definition, and indeed one that is applicable to a wide variety of tasks, amongst which falls modern-day LI. The archetypal text categorization task is perhaps the classification of newswire articles according to the topics that they discuss, exemplified by the Reuters-21578 dataset (Debole & Sebastiani, 2005). However, LI has particular characteristics that make it different from typical text categorization tasks:

1. Text categorization tends to use statistics about the frequency of words to model documents, but for LI purposes there is no universal notion of a *word*: LI must cater for languages where whitespace is not used to denote word boundaries. Furthermore, the determination of the appropriate word tokenization strategy for a given document presupposes knowledge of the language the document is written in, which is exactly what we assume we *don't* have access to in LI.

2. In text categorization tasks, the set of labels usually only applies to a particular dataset. For example, it is not meaningful to ask which of the Reuters-21578 labels is applicable to the abstract of a biomedical journal article. However, in LI there is a clear notion of language that is independent of domain: it is possible to recognize that a text is in English regardless of whether it is from a biomedical journal, a microblog post, or a newspaper article.

3. In LI, classes can be somewhat multi-modal, in that text in the same language can sometimes be written with different orthographies and stored in different encodings, but correspond to the same class.

4. In LI, labels are non-overlapping and mutually exclusive, meaning that a text can only be written in one language. This does not preclude the existence of multilingual documents which contain text in more than one language, but when this is the case, the document can always be uniquely divided into monolingual segments. This is in contrast to text categorization involving multi-labeled documents, where it is generally not possible to associate specific segments of the document with specific labels.

These distinguishing characteristics present unique challenges and offer particular opportunities, so much so that research in LI has generally proceeded independently of text categorization research.

2.1 An Archetypal Language Identifier

The design of a supervised language identifier can generally be deconstructed into four key steps:

1. A document representation is selected;

2. A language model of each of a pre-defined set of training languages is derived from a training corpus of labelled documents;

3. A function is defined that determines how well a given document fits the language model of each of the training languages;

4. The language of the document is predicted based on the best-fitting language model.

2.2 Previous Surveys

Although there are some dedicated survey articles, these tend to be relatively short; there have not been any comprehensive surveys of research in automated LI of text to date. The largest survey so far can be found in the literature review of Marco Lui's PhD thesis (Lui, 2014a), which served as an early draft and starting point for the current article. Zampieri (2016) provides a historical overview of language identification focusing on the use of n-gram language models. Qafmolla (2017) gives a brief overview of some of the methods used for LI, and Garg, Gupta, and Jindal (2014) provide a review of some of the techniques and applications used previously. Shashirekha (2014) gives a short overview of some of the challenges, algorithms and available tools for LI. Juola (2006) provides a brief summary of LI, how it relates to other research areas, and some outstanding challenges, but only does so in general terms and does not go into any detail about existing work in the area. Another brief article about LI is Muthusamy and Spitz (1997), which covers LI both of spoken language as well as of written documents, and also discusses LI of documents stored as images rather than digitally-encoded text.

3. A Brief History of LI

LI as a task predates computational methods – the earliest interest in the area was motivated by the needs of translators, and simple manual methods were developed to quickly identify documents in specific languages.

The earliest known work to describe a functional LI program for text is by Mustonen (1965), a statistician, who used multiple discriminant analysis to teach a computer how to distinguish, at the word level, between English, Swedish and Finnish. Mustonen compiled a list of linguistically-motivated character-based features, and trained his language identifier on 300 words for each of the three target languages. The training procedure created two discriminant functions, which were tested with 100 words for each language. The experiment resulted in 76% of the words being correctly classified; even by current standards this percentage would be seen as acceptable given the small amount of training material, although the composition of training and test data is not clear, making the experiment unreproducible.

Gold (1967) sought to investigate language learnability from the perspective of formal language theory, and thus formalized LI as a closed-class classification problem and investigated theoretical limitations on the ability to separate languages from different classes. In practice, the definition of language of Gold (1967) is only tangentially related to natural language, but the results of his research are much more general: the key result is that given sufficient labeled data in multiple languages (*language* as in *language theory* rather than

natural language), it is possible to construct an algorithm that will correctly distinguish context-free languages. The proof given is a proof of existence; it does not actually suggest how to construct such an algorithm. In short, the results of Gold (1967) prove that (assuming documents are generated using a context-free grammar), there exists an algorithm to perform supervised text categorization that will converge to the correct answer in a finite number of steps. This may be a significant theoretical result, but it is a result that is generally taken for granted in any sort of supervised machine learning applied to text classification.

In the early 1970s, Nakamura (1971) considered the problem of automatic LI. According to Rau (1974) and the available abstract of Nakamura's article,[2] his language identifier was able to distinguish between 25 languages written with the Latin alphabet. As features, the method used the occurrence rates of characters and words in each language. From the abstract it seems that, in addition to the frequencies, he used some binary presence/absence features of particular characters or words, based on manual LI.

Rau (1974) wrote his master's thesis "Language Identification by Statistical Analysis" for the Naval Postgraduate School at Monterey, California. The continued interest and the need to use LI of text in military intelligence settings is evidenced by the recent articles of, for example, Rafidha Rehiman, Keerthy, Lakshmi, and Sreekumar (2013), Rowe, Schwamm, and Garfinkel (2013), Tratz (2014), and Voss, Tratz, Laoudi, and Briesch (2014). As features for LI, Rau (1974) used, e.g., the relative frequencies of characters and character bigrams. With a majority vote classifier ensemble of seven classifiers using Kolmogor-Smirnov's Test of Goodness of Fit and Yule's characteristic (K), he managed to achieve 89% accuracy over 53 characters when distinguishing between English and Spanish. His thesis actually includes the identifier program code (for the IBM System/360 Model 67 mainframe), and even the language models in printed form.

Much of the earliest work on automatic LI was focused on identification of spoken language, or did not make a distinction between written and spoken language. For example, the work of House and Neuburg (1977) is primarily focused on LI of spoken utterances, but makes a broader contribution in demonstrating the feasibility of LI on the basis of a statistical model of broad phonetic information. However, their experiments do not use actual speech data, but rather "synthetic" data in the form of phonetic transcriptions derived from written text.

Another subfield of speech technology, speech synthesis, has also generated a considerable amount of research in the LI of text, starting from the 1980s. In speech synthesis, the need to know the source language of individual words is crucial in determining how they should be pronounced. Church (1985) uses the relative frequencies of character trigrams as probabilities and determines the language of words using a Bayesian model. Church explains the method – that has since been widely used in LI – as a small part of an article concentrating on many aspects of letter stress assignment in speech synthesis, which is probably why Beesley (1988) is usually attributed to being the one to have introduced the aforementioned method to LI of text. As Beesley's article concentrated solely on the problem of LI, this single focus probably enabled his research to have greater visibility. The role of the program implementing his method was to route documents to MT systems,

2. We were unable to obtain the original article, so our account of the paper is based on the abstract and reports in later published articles.

and Beesley's paper more clearly describes what has later come to be known as a character n-gram model. The fact that the distribution of characters is relatively consistent for a given language was already well known (Zipf, 1932).

The highest-cited early work on automatic LI is Cavnar and Trenkle (1994). Cavnar and Trenkle's method (which we describe in detail in Section 6.6) builds up per-document and per-language profiles, and classifies a document according to which language profile it is most similar to, using a rank-order similarity metric. They evaluate their system on 3478 documents in eight languages obtained from USENET newsgroups, reporting a best overall LI accuracy of 99.8%. Gertjan van Noord produced an implementation of the method of Cavnar and Trenkle named `TextCat`, which has become eponymous with the method itself (van Noord, 1997). `TextCat` is packaged with pre-trained models for a number of languages, and so it is likely that the strong results reported by Cavnar and Trenkle, combined with the ready availability of an "off-the-shelf" implementation, has resulted in the exceptional popularity of this particular method. Cavnar and Trenkle (1994) can be considered a milestone in automatic LI, as it popularized the use of automatic methods on character n-gram models for LI, and to date the method is still considered a benchmark for automatic LI.

4. On Notation

This section introduces the notation used throughout this article to describe LI methods. We have translated the notation in the original papers to our notation, to make it easier to see the similarities and differences between the LI methods presented in the literature. The formulas presented could be used to implement language identifiers and re-evaluate the studies they were originally presented in.

A corpus C consists of individual tokens u which may be bytes, characters or words. C is comprised of a finite sequence of individual tokens, $u_1, ..., u_{l_C}$. The total count of individual tokens u in C is denoted by l_C. In a corpus C with non-overlapping segments, each segment is referred to as C_s, which may be a short document or a word or some other way of segmenting the corpus. The number of segments is denoted as l_S.

A feature f is some countable characteristic of the corpus C. When referring to the set of all features F in a corpus C, we use C^F, and the number of features is denoted by l_{C^F}. A set of unique features in a corpus C is denoted by $U(C)$. The number of unique features is referred to as $|U(C)|$. The count of a feature f in the corpus C is referred to as $c(C, f)$. If a corpus is divided into segments, the count of a feature f in C is defined as the sum of counts over the segments of the corpus, i.e. $c(C, f) = \sum_{s=1}^{l_S} c(C_s, f)$. Note that the segmentation may affect the count of a feature in C as features do not cross segment borders.

A frequently-used feature is an n-gram, which consists of a contiguous sequence of n individual tokens. An n-gram starting at position i in a corpus segment is denoted $u_{i,...,i+n-1}$, where positions $i+1, ..., i+n-1$ remain within the same segment of the corpus as i. If $n = 1$, f is an individual token. When referring to all n-grams of length n in a corpus C, we use C^n and the count of all such n-grams is denoted by l_{C^n}. The count of an n-gram f in a corpus segment C_s is referred to as $c(C_s, f)$ and is defined by Equation 1:

$$c(C_s, f) = \sum_{i=1}^{l_{C_s}+1-n} \begin{cases} 1 & \text{, if } f = u_{i,\ldots,i-1+n} \\ 0 & \text{, otherwise} \end{cases} \quad (1)$$

The set of languages is G, and l_G denotes the number of languages. A corpus C in language g is denoted by C_g. A language model O based on C_g is denoted by $O(C_g)$. The features given values by the model $O(C_g)$ are the domain $dom(O(C_g))$ of the model. In a language model, a value v for the feature f is denoted by $v_{C_g}(f)$. For each potential language g of a corpus C in an unknown language, a resulting score $R(g, C)$ is calculated. A corpus in an unknown language is also referred to as a test document.

5. Features

In this section, we present an extensive list of features used in LI, some of which are not self-evident. The equations written in the unified notation defined earlier show how the values v used in the language models are calculated from the tokens u. For each feature type, we generally introduce the first published article that used that feature type, as well as more recent articles where the feature type has been considered.

5.1 Bytes and Encodings

In LI, text is typically modeled as a stream of characters. However, there is a slight mismatch between this view and how text is actually stored: documents are digitized using a particular encoding, which is a mapping from characters (e.g. a character in an alphabet), onto the actual sequence of bytes that is stored and transmitted by computers. Encodings vary in how many bytes they use to represent each character. Some encodings use a fixed number of bytes for each character (e.g. ASCII), whereas others use a variable-length encoding (e.g. UTF-8). Some encodings are specific to a given language (e.g. GuoBiao 18030 or Big5 for Chinese), whereas others are specifically designed to represent as many languages as possible (e.g. the Unicode family of encodings). Languages can often be represented in a number of different encodings (e.g. UTF-8 and Shift-JIS for Japanese), and sometimes encodings are specifically designed to share certain codepoints (e.g. all single-byte UTF-8 codepoints are exactly the same as ASCII). Most troubling for LI, isomorphic encodings can be used to encode different languages, meaning that the determination of the encoding often doesn't help in honing in on the language. Infamous examples of this are the ISO-8859 and EUC encoding families. Encodings pose unique challenges for practical LI applications: a given language can often be encoded in different forms, and a given encoding can often map onto multiple languages.

Some LI research has included an explicit encoding detection step to resolve bytes to the characters they represent (Kikui, 1996), effectively transcoding the document into a standardized encoding before attempting to identify the language. However, transcoding is computationally expensive, and other research suggests that it may be possible to ignore encoding and build a single per-language model covering multiple encodings simultaneously (Kruengkrai, Srichaivattana, Sornlertlamvanich, & Isahara, 2005; Baldwin & Lui, 2010a). Another solution is to treat each language-encoding pair as a separate category (Cowie, Ludovik, & Zacharski, 1999; Suzuki, Mikami, Ohsato, & Chubachi, 2002; Singh & Gorla,

2007; Brown, 2012). The disadvantage of this is that it increases the computational cost by modeling a larger number of classes. Most of the research has avoided issues of encoding entirely by assuming that all documents use the same encoding (Mandl, Shramko, Tartakovski, & Womser-Hacker, 2006). This may be a reasonable assumption in some settings, such as when processing data from a single source (e.g. all data from Twitter and Wikipedia is UTF-8 encoded). In practice, a disadvantage of this approach may be that some encodings are only applicable to certain languages (e.g. S-JIS for Japanese and Big5 for Chinese), so knowing that a document is in a particular encoding can provide information that would be lost if the document is transcoded to a universal encoding such as UTF-8. Li and Momoi (2001) used a parallel state machine to detect which encoding scheme a file could potentially have been encoded with. The knowledge of the encoding, if detected, is then used to narrow down the possible languages.

Most features and methods do not make a distinction between bytes or characters, and because of this we will present feature and method descriptions in terms of characters, even if byte tokenization was actually used in the original research.

5.2 Characters

In this section, we review how individual character tokens have been used as features in LI.

Non-alphabetic or non-ideographic characters Ranaivo-Malançon and Ng (2005) used the formatting of numbers when distinguishing between Malay and Indonesian. King and Abney (2013) used the presence of non-alphabetic characters between the current word and the words before and after as features. Elfardy and Diab (2013) used emoticons (or emojis) in Arabic dialect identification with Naive Bayes ("NB"; see Section 6.5). Simaki, Simakis, Paradis, and Kerren (2017) used the frequencies of punctuation and certain special symbols as features when distinguishing between English variants. They used the Relief feature selection algorithm (Kira & Rendell, 1992) to rank the features, and found the symbol and punctuation characters to rank 5th and 9th, respectively, out of the 31 ranked features.

Alphabets Henrich (1989) used knowledge of alphabets to exclude languages where a language-unique character in a test document did not appear. Giguet (1995) used alphabets collected from dictionaries to check if a word might belong to a language. Hanif, Latif, and Khiyal (2007) used the Unicode database to get the possible languages of individual Unicode characters. Lately, the knowledge of relevant alphabets has been used for LI also by Samih (2017) and Hasimu and Silamu (2017). Hasimu and Silamu (2017) used the knowledge of alphabets as the first part of a staged LI scheme, determining the possible language groups based on the alphabet used as well as removing possible content in other scripts from sentences including code-switching.[3] Then they proceeded to identify the language group and the exact language in later stages.

Capitalization Capitalization is mostly preserved when calculating character n-gram frequencies, but in contexts where it is possible to identify the orthography of a given document and where capitalization exists in the orthography, lowercasing can be used to

3. See the recent survey by Sitaram, Chandu, Rallabandi, and Black (2019) for information on computational approaches to code-switching.

reduce sparseness. In recent LI work, capitalization was used as a special feature by Basile et al. (2017), Bestgen (2017), and Simaki et al. (2017).

The number of characters in words and word combinations Langer (2001) was the first to use the length of words in LI. Nobesawa and Tahara (2005) used the length of full person names comprising several words. Lately, the number of characters in words has been used for LI by Dongen (2017), van der Lee and Bosch (2017), Samih (2017), and Simaki et al. (2017). Dongen (2017) also used the length of the two preceding words.

The frequency or probability of each character Kerwin (2006) used character frequencies as feature vectors. In a feature vector, each feature f has its own integer value. The raw frequency – also called term frequency (TF) – is calculated for each language g as:

$$v_{C_g}(f) = c(C_g, f) \tag{2}$$

Rau (1974) was the first to use the probability of characters. He calculated the probabilities as relative frequencies, by dividing the frequency of a feature found in the corpus by the total count of features of the same type in the corpus. When the relative frequency of a feature f is used as a value, it is calculated for each language g as:

$$v_{C_g}(f) = \frac{c(C_g, f)}{l_{C_g^F}} \tag{3}$$

Tran and Sharma (2005) calculated the relative frequencies of one-character prefixes, and Windisch and Csink (2005) did the same for one-character suffixes.

Ng and Selamat (2009) calculated character frequency document frequency ("LFDF") values. Takçı and Güngör (2012) compared their own Inverse Class Frequency ("ICF") method with the Arithmetic Average Centroid ("AAC") and the Class Feature Centroid ("CFC") feature vector updating methods. They performed experiments on nine languages using the ECI/MC1 corpus (Armstrong-Warwick, Thompson, McKelvie, & Petitpierre, 1994), and found ICF to clearly outperform both the AAC and CFC updating methods.[4] In ICF a character appearing frequently only in some language gets more positive weight for that language. The values differ from Inverse Document Frequency ("IDF", Equation 8), as they are calculated using also the frequencies of characters in other languages. Their ICF-based vectors generally performed better than those based on AAC or CFC. Takçı and Ekinci (2012) explored using the relative frequencies of characters with similar discriminating weights. Takçı and Güngör (2012) also used Mutual Information ("MI") and chi-square weighting schemes with characters.

Baldwin and Lui (2010a) compared the identification results of single characters with the use of character bigrams and trigrams when classifying over 67 languages. Both bigrams and trigrams generally performed better than unigrams. Jauhiainen (2010) also found that the identification results from identifiers using just characters are generally worse than those using character sequences.

4. For 100 character test documents ICF achieved an average F-score of 0.973, while AAC got 0.889, and CFC 0.672.

5.3 Character Combinations

In this section we consider the different combinations of characters used in the literature. Character n-grams mostly consist of all possible characters in a given encoding, but can also consist of only alphabetic or ideographic characters.

Co-occurrences Windisch and Csink (2005) calculated the co-occurrence ratios of any two characters, as well as the ratio of consonant clusters of different sizes to the total number of consonants. Sterneberg (2012) used the combination of every bigram and their counts in words. van der Lee and Bosch (2017) used the proportions of question and exclamation marks to the total number of the end of sentence punctuation as features with several machine learning algorithms.

Franco-Salvador, Plotnikova, Pawar, and Benajiba (2017b) used FastText to generate character n-gram embeddings (Joulin, Grave, Bojanowski, & Mikolov, 2017). Neural network generated embeddings are explained in Section 5.6.

Vowel–consonant relationship Rau (1974) used the relative frequencies of vowels following vowels, consonants following vowels, vowels following consonants and consonants following consonants. Dongen (2017) used vowel–consonant ratios as one of the 30 features with Support Vector Machines ("SVMs", Section 6.8), Decision Trees ("DTs", Section 6.2), and Conditional Random Fields ("CRFs", Section 10.7). The vowel–consonant ratio ranked 14th in her analysis of the predictive strength of the features she used.

Character repetition Elfardy and Diab (2013) used the existence of word lengthening effects and repeated punctuation as features. Banerjee et al. (2014) used the presence of characters repeating more than twice in a row as a feature with simple scoring (Equation 17). Barman, Das, Wagner, and Foster (2014a) used more complicated repetitions identified by regular expressions. Sikdar and Gambäck (2016) used letter and character bigram repetition with a CRF. Martinc, Škrjanec, Zupan, and Pollak (2017) used the count of character sequences with three or more identical characters ("character flood counts") as part of their feature selection with several machine learning algorithms. In the "PAN" (Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection workshop) 2017 language variety identification shared task, they attained 3rd position out of 22 participating teams (Rangel, Rosso, Potthast, & Stein, 2017b). They used FeatureUnion in scikit-learn (Pedregosa et al., 2011) to determine the weights for different types of features, and found character flood counts to rank higher than POS or punctuation trigrams and word bigrams.

n-grams of characters of the same size Character n-grams are continuous sequences of characters of length n. They can be either consecutive or overlapping. Consecutive character bigrams created from the four character sequence *door* are *do* and *or*, whereas the overlapping bigrams are *do*, *oo*, and *or*. Overlapping n-grams are most often used in the literature. Overlapping produces a greater number and variety of n-grams from the same amount of text.

Rau (1974) was the first to use combinations of any two characters. He calculated the relative frequency of each bigram. Table 2 lists more recent articles where relative frequencies of n-grams of characters have been used. The table shows how different lengths of n-grams fared if they were compared with each other. The only conclusions that can be

Article	1	2	3	4	5	6	7
Bošnjak, Rodrigues, and Sarmento (2013)				***			
Zampieri, Gebre, and Diwersy (2013)		*	**	***	*		
Das and Gambäck (2014)	*	*	**	**	*	*	*
Indhuja, Indu, Sreejith, and Reghu Raj (2014)	***						
Ljubešić and Kranjcić (2014)			*			*	
Minocha and Tyers (2014)	*	*	***	*	*		
Pethö and Mózes (2014)		*	*	**	***		
Sadat, Kazemi, and Farzindar (2014a, 2014b)	*	***	*				
Tan, Zampieri, Ljubešić, and Tiedemann (2014)		*	*	*	***	**	
Zaidan and Callison-Burch (2014)	*		**		***		
Zampieri and Gebre (2014)			***				
Franco-Salvador, Rangel, Rosso, Taulé, and Martít (2015a)				***			
Jauhiainen, Jauhiainen, and Lindén (2015a)	*	*	*	*	*	***	**...
King, Kübler, and Hooper (2015)					***		
Panich (2015)	*	*	*	*	*	*	
Zampieri, Gebre, Costa, and Genabith (2015)					***		
Abainia, Ouamour, and Sayoud (2016)	*	*	*				
Castro, Souza, and de Oliveira (2016), Castro, Souza, Vitório, Santos, and Oliveira (2017)		*	*	*	*	***	**
Giwa (2016)	*	*	**	***	*	*	*
Hanani, Qaroush, and Taylor (2016)			***				
Duvenhage, Ntini, and Ramonyai (2017)					***		
Jourlin (2017)	***						

Table 2: List of articles (2013–2017) where relative frequencies of character n-grams have been used as features. The columns indicate the length of the n-grams used. "***" indicates the empirically best n-gram length in that paper, and "**" the second-best n-gram length. "*" indicates less effective n-gram lengths. "..." indicates that even higher n-grams were used.

drawn from the results is that if character n-grams longer than one were used they usually performed better than unigrams, and character 7-grams performed worse than shorter lengths. The relative performance of different n-gram lengths is hugely dependent on the overall setting in which the evaluation was done.

Rau (1974) also used the relative frequencies of two character combinations which had one unknown character between them, also known as gapped bigrams. As an example, gapped bigrams from the word *known* would be *ko*, *nw*, and *on*. Seifart and Mundry (2015) used a modified relative frequency of character unigrams and bigrams.

Character trigram frequencies relative to the word count were used by Vega and Bressan (2001a), who calculated the values $v_C(f)$ as in Equation 4. Let T be the word-tokenized segmentation of the corpus C of character tokens, then:

$$v_C(f) = \frac{c(C, f)}{l_T} \tag{4}$$

Article	2	3	4	5	6	7	8
Ramisch (2008)	**	**	**	***			
You, Chen, Chu, Soong, and Wang (2008)		**	***	**	*	*	
Stupar, Jurić, and Ljubešić (2011), Tiedemann and Ljubešić (2012)		***					
Goldszmidt, Najork, and Paparizos (2013)		***					
Bar and Dershowitz (2014)		***					
Brown (2014b)		***					
Gamallo, Garcia, and Sotelo (2014)		***					
Hurtado, Pla, Giménez, and Sanchis (2014)			***				
Indhuja et al. (2014)	*	*					
Leidig (2014)				***			
Mendizabal, Carandell, and Horowitz (2014)	*	*	**	***	**	**	**
Pethö and Mózes (2014)	*	*	**	***			
Sadat et al. (2014a, 2014b)	***	**					
Ullman (2014)			***				
Cianflone and Kosseim (2016)	*	*	*	*	*	**	***
Martadinata, Trisedya, Manurung, and Adriani (2016)		*	**	***			
Samih and Maier (2016), Samih (2017)				***			

Table 3: List of recent articles where Markovian character n-grams have been used as features. The columns indicate the length of the n-grams used. "***" indicates the best and "**" the second-best n-gram length as reported in the article in question. "*" indicates less effective n-gram lengths.

where $c(C, f)$ is the count of character trigrams f in C, and l_T is the total word count in the corpus. Later n-gram frequencies relative to the word count were used by Hamzah (2010) for character bigrams and trigrams.

House and Neuburg (1977) divided characters into five phonetic groups and used a Markovian method to calculate the probability of each bigram consisting of these phonetic groups. In Markovian methods, the probability of a given character u_i is calculated relative to a fixed-size character context $u_{i-n+1}, ..., u_{i-1}$ in corpus C, as follows:

$$P(u_i|u_{i-n+1,...,i-1}) = \frac{c(C, u_{i-n+1,...,i})}{c(C, u_{i-n+1,...,i-1})} \tag{5}$$

where $u_{i-n+1,...,i-1}$ is an n-gram prefix of $u_{i-n+1,...,i}$ of length $n-1$. In this case, the probability $P(u_i|u_{i-n+1,...,i-1})$ is the value $v_C(f)$, where $f = u_{i-n+1,...,i}$, in the model $O(C)$. Ludovik and Zacharski (1999) used 4-grams with recognition weights which were derived from Markovian probabilities. Table 3 lists some of the more recent articles where Markovian character n-grams have been used. Markovian character 4- and 5-grams seem to perform well in comparison to other lengths, except in the evaluations of Cianflone and Kosseim (2016) where 8-grams fared the best.

Vitale (1991) was the first author to propose a full-fledged probabilistic language identifier. He defines the probability of a trigram f being written in the language g to be:

$$P(g|f) = \frac{P(f|g)P(g)}{\sum_{h \in G} P(f|h)P(h)} \tag{6}$$

He considers the prior probabilities of each language $P(g)$ to be equal, which leads to:

$$P(g|f) = \frac{P(f|g)}{\sum_{h \in G} P(f|h)} \tag{7}$$

Vitale (1991) used the probabilities $P(g|f)$ as the values $v_{C_g}(f)$ in the language models.

MacNamara, Cunningham, and Byrne (1998) used a list of the most frequent bigrams and trigrams with logarithmic weighting. Chanda, Das, and Mazumdar (2016b) used a sorted dictionary composed of the 400 most common character 4-grams with the out-of-place method (Section 6.6) in word level language identification of English–Bengali code-mixed text. Plaza Cagigós (2017) used a dictionary of character 3-grams.

Prager (1999) was the first to use direct frequencies of character n-grams as feature vectors. He compared the use of character n-grams from 2 to 5 with the use of words. Using words resulted in better identification results than using character bigrams (test document sizes of 20, 50, 100 or 200 characters), but always worse than character 3-, 4- or 5-grams. However, the combined use of words and character 4-grams gave the best results of all tested combinations, obtaining 95.6% accuracy for 50 character sequences when choosing between 13 languages.

Vinosh Babu and Baskaran (2005) used Principal Component Analysis ("PCA") to map the bigrams in feature vectors representing languages onto a lower-dimensional space, thus up-weighting the most discriminant features. Murthy and Kumar (2006) used the most frequent and discriminating byte unigrams, bigrams, and trigrams among their feature functions. They define the most discriminating features as those which have the most differing relative frequencies between the models of the different languages. Gottron and Lipka (2010) tested n-grams from two to five using frequencies as feature vectors, frequency ordered lists, relative frequencies, and Markovian probabilities. Adouane (2016) evaluated different length character n-gram features with SVMs in Arabic variety identification. She found that 5-grams were best with 88.74% accuracy, with 6-grams not far behind at 88.61%. The combined use of 5- and 6-grams, however, fared even better gaining 89.02% accuracy. Table 4 lists the more recent articles where the frequency of character n-grams have been used as features. In the method column, "RF" refers to Random Forest (cf. Section 6.2), "LR" to Logistic Regression (Section 6.7), "KRR" to Kernel Ridge Regression (Section 6.6), "KDA" to Kernel Discriminant Analysis (Section 6.6), and "NN" to Neural Networks (Section 6.9). Malmasi and Dras (2017) compare the performance of SVM classifiers using character n-grams of size one to six, as well as words, and found that character 6-grams perform the best on the dataset of the Discriminating between Similar Languages ("DSL") 2016 shared task (Malmasi et al., 2016).

Giguet (1995) used the last two and three characters of open class words. Suzuki et al. (2002) used an unordered list of distinct trigrams with the simple scoring method (Section 6.3). Hayati (2004) used Fisher's discriminant function to choose the 1000 most discriminating trigrams. Bilcu and Astola (2006) used unique 4-grams of characters with positive Decision Rules (Section 6.1). Ozbek, Rosenn, and Yeh (2006) used the frequencies of uni-, bi- and trigrams of characters in words unique to a language in a NB baseline language identifier. A NB classifier using morpheme and morphotactic features achieved a slight improvement over the baseline.

Article	1	2	3	4	5	6+	Method
Adouane (2016)	*	*	*	*	*	*	SVM (cf. §6.8)
Al-Badrashiny and Diab (2016)	*	*	*	*	*		CRF (cf. §10.7)
Alshutayri et al. (2016)	*	*	*				SVM, DT (cf. §6.2)
Barbaresi (2016)		*	*	*	*	*	NB (cf. §6.5), XGBoost (cf. §6.10), RF (cf. §6.2)
Ciobanu and Dinu (2016)	*	*	*	*			LR (cf. §6.7)
Giwa (2016)		*	*	*	*		SVM
Goutte and Léger (2016)						*	SVM, NB
Ionescu and Popescu (2016)		*	*	*	*	*	KRR (cf. §6.6), KDA (cf. §6.6)
Lamabam and Chakma (2016)			*				CRF
Criscuolo and Aluísio (2017)					*		NB
Malmasi and Dras (2017)	*	*	*	*	*	*	SVM
Malmasi (2017)				*			SVM
Mathur, Misra, and Budur (2017)	*	*	*	*	*	*	NB, LR
Oliveira and Neto (2017)			*	*	*	*	SVM
Plaza Cagigós (2017)			*				NB, SVM, DT, NN (cf. §6.9)
Rangel, Franco-Salvador, and Rosso (2017a)				*			SVM, NB, NN, ...
Schaetti (2017)		*					NN

Table 4: Recent papers (2016–) where the frequency of character n-grams has been used to generate feature vectors. The columns indicate the length of the n-grams used, and the machine learning method(s) used. The relevant section numbers for the methods are mentioned in parentheses. "..." indicates that even more methods were used.

Li and Momoi (2001) divided possible character bigrams into those that are commonly used in a language and to those that are not. They used the ratio of the commonly used bigrams to all observed bigrams to give a confidence score for each language. Xafopoulos, Kotropoulos, Almpanidis, and Pitas (2004) used the difference between the ISO Latin-1 code values of character bigrams as well as gapped character bigrams.

Artemenko and Shramko (2005) used the IDF and the transition probability of trigrams. They calculated the IDF values $v_{C_g}(f)$ of trigrams f for each language g, as in Equation 8, where $c(C_g, f)$ is the number of trigrams f in the corpus of the language g and $df(C_G, f)$ is the number of languages in which the trigram f is found, where C_G is the language-segmented training corpus with each language as a single segment.

$$v_{C_g}(f) = \frac{c(C_g, f)}{df(C_G, f)} \tag{8}$$

df is defined as:

$$df(C_G, f) = \sum_{g \in G} \begin{cases} 1 & \text{, if } c(C_g, f) > 0 \\ 0 & \text{, otherwise} \end{cases} \tag{9}$$

Malmasi, Refaee, and Dras (2015) used n-grams from one to four, which were weighted with "TF-IDF" (Term Frequency–Inverse Document Frequency). TF-IDF was calculated as:

$$v_{C_g}(f) = c(C_g, f) \log \frac{l_G}{df(C_G, f)} \tag{10}$$

TF-IDF weighting or close variants have been widely used for LI. Thomas and Verma (2007) used "CF-IOF" (Class Frequency-Inverse Overall Frequency) weighted 3- and 4-grams.

Jhamtani, Bhogi, and Raychoudhury (2014) used the logarithm of the ratio of the counts of character bigrams and trigrams in the English and Hindi dictionaries as one of the features in word level language identification in code-switched texts. Zamora, Bruzón, and Bueno (2014) used a feature weighting scheme based on mutual information ("MI"). They also tried weighting schemes based on the "GSS" (Galavotti, Sebastiani, and Simi) and "NGL" (Ng, Goh, and Low) coefficients, but using the MI-based weighting scheme proved the best in their evaluations when they used the sum of values method (Equation 18). Martinc et al. (2017) used punctuation trigrams as one of their features with logistic regression. In punctuation trigrams, the first character has to be a punctuation mark (but not the other two characters). Saharia (2017) used consonant bi- and trigrams which were generated from words after the vowels had been removed in identifying the language of individual words in English–Assamese mixed-language social network messages.

Character n-grams of differing sizes The language models mentioned earlier consisted only of n-grams of the same size n. If n-grams from one to four were used, then there were four separate language models. Cavnar and Trenkle (1994) created ordered lists of the most frequent n-grams for each language and used them with the rank-order method (see Section 6.6). Singh and Goyal (2014) used similar n-gram lists with symmetric cross-entropy in a tweet language identification shared task (Zubiaga et al., 2014), but their results were inferior to the best results of the other six participating teams.

Russell and Lapalme (2003) used a Markovian method to calculate the probability of byte trigrams interpolated with byte unigrams. Their identifier reached a precision of 72% for 10 character test texts when distinguishing between 27 languages. Vatanen, Väyrynen, and Virpioja (2010) created a language identifier based on character n-grams of different sizes over 281 languages, and obtained an identification accuracy of 62.8% for extremely short samples (5–9 characters). Their language identifier was used or evaluated by Rodrigues (2012), Maier and Gómez-Rodríguez (2014), and Jauhiainen, Lindén, and Jauhiainen (2017b). Rodrigues (2012) managed to improve the identification results by feeding the raw language distance calculations into an SVM.

Kikui (1996) used up to the four last characters of words and calculated their relative frequencies. Ahmed, Cha, and Tappert (2004) used frequencies of 2–7-grams, normalized relative to the total number of n-grams in all the language models as well as the current language model. Jauhiainen (2010) compared the use of different sizes of n-grams in differing combinations, and found that combining n-grams of differing sizes resulted in better identification scores. Lui and Baldwin (2011, 2012, 2014) used mixed length domain-independent language models of byte n-grams from one to three or four.

Mixed length language models were also generated by Brown (2012) and later by Brown (2013, 2014b), who used the most frequent and discriminating n-grams longer than two

bytes, up to a maximum of 12 bytes, based on their weighted relative frequencies. K of the most frequent n-grams were extracted from training corpora for each language, and their relative frequencies were calculated. In the tests reported in (Brown, 2013), K varied from 200 to 3,500 n-grams.

Stensby, Oommen, and Granmo (2010) used mixed-order n-gram frequencies relative to the total number of n-grams in the language model. Sterneberg (2012) used frequencies of n-grams from one to five and gapped 3- and 4-grams as features with an SVM. King et al. (2014) used character n-grams as a backoff from Markovian word n-grams. Shrestha (2014) used the frequencies of word initial n-grams ranging from 3 to the length of the word minus 1. Ács, Grad-Gyenge, Bruno, and Oliveira (2015) used the most relevant n-grams selected using the absolute value of the Pearson correlation. Mandal, Banerjee, Naskar, Rosso, and Bandyopadhyay (2015) used only the first 10 characters from a longer word to generate the n-grams, while the rest were ignored. Qiao and Lévy (2015) used only those n-grams which had the highest TF-IDF scores. Bestgen (2017) used character n-grams weighted by means of the "BM25" (Best Match 25) weighting scheme.

Table 5 lists recent articles where character n-grams of differing sizes have been used. "LR" in the methods column refers to Logistic Regression (Section 6.6), "LSTM RNN" to Long Short-Term Memory Recurrent Neural Networks (Section 6.9), and "DAN" to Deep Averaging Networks (Section 6.9). Hanani, Qaroush, and Taylor (2017) used byte n-grams up to length 25. In the evaluations of Jauhiainen et al. (2017b), the method of Vatanen et al. (2010) was the best method for texts of 20 characters or less when distinguishing between 285 languages in an out-of-domain setting. Malmasi and Zampieri (2017a) used a Random Forest meta-classifier ensemble to combine the results of separate SVMs using character n-grams from one to eight, words, and i-Vectors in Arabic Dialect Identification (ADI) shared task at the VarDial Evaluation Campaign 2017 (Zampieri et al., 2017). Sanchez-Perez, Markov, Gómez-Adorno, and Grigori (2017) evaluated different combinations of character n-grams as well as their combinations with words in discriminating between Spanish variants using the dataset from the 2016 edition of the DSL shared task (Malmasi et al., 2016). They used Liblinear SVM in all of their experiments (Fan, Chang, Hsieh, Wang, & Lin, 2008). Their evaluations indicate that higher order character n-grams perform better than lower order n-grams. Their best results were achieved using character n-grams of length two to eight, together with word unigrams and bigrams.

Consonant or vowel sequences Sterneberg (2012) used consonant sequences generated from words as one of the features in a SVM classifier. He does not explain how important the consonant sequences were when compared with the other features he used such as character n-grams and words. Anand (2014) used the presence of vowel sequences as a feature with a NB classifier (see Section 6.5) when distinguishing between English and transliterated Indian languages.

Article	1	2	3	4	5	6	7	8+	Method
Adouane, Semmar, and Johansson (2016a, 2016c)	*	*	*						SVM (cf. §6.8)
Adouane, Semmar, and Johansson (2016b), Adouane, Semmar, Johansson, and Bobicev (2016d)					*	*			SVM
Balažević, Braun, and Müller (2016)	*	*	*	*					Product (cf. §6.5), SVM, LR (cf. §6.7), Sum (cf. §6.4)
Çöltekin and Rama (2016)	*	*	*	*	*	*			SVM
Eldesouki, Dalvi, Sajjad, and Darwish (2016)		*	*	*	*				SVM, LR, NN (cf. §6.9), NB (cf. §6.5)
He, Zhang, Zhao, Li, and Yan (2016)	*	*	*	*					LR (cf. §6.6)
Jauhiainen, Lindén, and Jauhiainen (2016)	*	*	*	*	*	*	*	*	HeLI (cf. §6.5)
Malmasi and Zampieri (2016, 2017b)	*	*	*	*	*	*			SVM
Piergallini, Shirvani, Gautam, and Chouikha (2016a)	*	*	*	*					LR
Piergallini, Shirvani, Gautam, and Chouikha (2016b)	*	*	*						LR
Radford and Gallé (2016)		*	*	*	*				LR
Samih, Maharjan, Attia, Kallmeyer, and Solorio (2016)		*	*						LSTM RNN (cf. §6.9)
Xu, Wang, and Li (2016)	*	*	*						SVM
Alrifai, Rebdawi, and Ghneim (2017)		*	*	*	*	*	*		SVM
Barbaresi (2017)		*	*	*	*	*	*		NB
Bestgen (2017)	*	*	*	*	*	*	*	*	SVM
Clematide and Makarov (2017)	*	*	*	*	*	*			NB, CRF (cf. §10.7), SVM
Espichán-Linares and Oncevay-Marcos (2017)		*	*						SVM, NB, ...
Franco-Salvador et al. (2017b)			*	*	*	*			DAN (cf. §6.9)
Gamallo, Pichel, and Alegria (2017)					*	*	*		Perplexity (cf. §6.6)
Gómez-Adorno, Markov, Baptista, Sidorov, and Pinto (2017)			*	*	*				NB, SVM
Hanani et al. (2017)	*	*	*						SVM, NB, LR, DT (cf. §6.2)
Jauhiainen, Lindén, and Jauhiainen (2017a)	*	*	*	*	*	*	*	*	HeLI
Jauhiainen et al. (2017b)	*	*	*	*	*	*			HeLI
Malmasi and Dras (2017)	*	*	*	*	*	*			SVM
Malmasi and Zampieri (2017a)	*	*	*	*	*	*	*	*	SVM
Mathur et al. (2017)		*	*	*	*				RNN
Miura, Taniguchi, Taniguchi, Misawa, and Ohkuma (2017)	*	*	*	*	*	*			SVM
Sanchez-Perez et al. (2017)			*	*	*	*	*	*	SVM
Tellez, Miranda-Jiménez, Graff, and Moctezuma (2017)	*		*		*		*	*	SVM
Espichán-Linares and Oncevay-Marcos (2018)		*	*	*					SVM, NB, ...

Table 5: List of articles (2016-) where character n-grams of differing sizes have been used as features. The numbered columns indicate the length of the n-grams used. The method column indicates the method used with the n-grams. The relevant section numbers are mentioned in parentheses. "..." indicates that even more methods were used.

Unique character combinations Henrich (1989) and Vitale (1991) used character combinations (of different sizes) that either existed in only one language or did not exist in one or more languages. Vitale (1991) used them with Decision Rules (Section 6.1) in determining the etymological grouping of proper names. An additional trigram analysis phase was necessary for those names not identified by Decision Rules.

5.4 Morphemes, Syllables and Chunks

Morphemes Giguet (1998) used the suffixes of lexical words derived from untagged corpora. El-Shishiny et al. (2004) used prefixes and suffixes determined using linguistic knowledge of the Arabic language. Marcadet, Fischer, and Waast-Richard (2005) used suffixes and prefixes in rule-based LI. Ozbek et al. (2006) used morphemes and morpheme trigrams (morphotactics) constructed by Creutz's algorithm (Creutz, 2003). Hammarström (2007) used prefixes and suffixes constructed by his own algorithm, which was later also used by Ceylan and Kim (2009). Romsdorfer and Pfister (2007) used morpheme lexicons in LI. Ceylan and Kim (2009) compared the use of morphological features with the use of variable sized character n-grams. When choosing between ten European languages, the morphological features obtained only 26.0% accuracy while the n-grams reached 82.7%. Yeong and Tan (2010) lemmatized Malay words in order to get the base forms. Lu and Mohamed (2011) used a morphological analyzer of Arabic. Zampieri et al. (2013) used morphological information from a part-of-speech (POS) tagger. Anand (2014) and Banerjee et al. (2014) used manually selected suffixes as features. Bekavac, Kocijan, and Tadić (2014) created morphological grammars to distinguish between Croatian and Serbian. Darwish, Sajjad, and Mubarak (2014) used morphemes created by Morfessor, but they also used manually created morphological rules. Gamallo et al. (2014) used a suffix module containing the most frequent suffixes. Dutta, Saha, Banerjee, and Naskar (2015) and Mandal et al. (2015) used word suffixes as features with CRFs. Barbaresi (2016) used an unsupervised method to learn morphological features from training data. The method collects candidate affixes from a dictionary built using the training data. If the remaining part of a word is found from the dictionary after removing a candidate affix, the candidate affix is considered to be a morpheme. Barbaresi (2016) used 5% of the most frequent affixes in language identification. Gómez-Adorno et al. (2017) used character n-grams classified into different types, which included prefixes and suffixes. Table 6 lists some of the more recent articles where prefixes and suffixes collected from a training corpus has been used for LI.

Syllables and syllable n-grams Chen, You, Chu, Zhao, and Wang (2006) used trigrams composed of syllables. Yeong and Tan (2010) used Markovian syllable bigrams for LI between Malay and English. Later Yeong and Tan (2011) also experimented with syllable uni- and trigrams. Murthy and Kumar (2006) used the most frequent as well as the most discriminating Indian script syllables, called aksharas. They used single aksharas, akshara bigrams, and akshara trigrams. Syllables would seem to be especially apt in situations where distinction needs to be made between two closely-related languages.

Chunks, chunk n-grams and n-grams of n-grams Chunks are certain combinations of characters that are deemed to have a special meaning in a language, such as syllables (You

Reference	1	2	3	4	Method
He et al. (2016)	*	*	*		LR (cf. §6.6)
Piergallini et al. (2016a)	*	*	*	*	LR
Samih and Maier (2016), Samih (2017)	*	*	*		CRF (cf. §10.7)
Schulz and Keller (2016)	*	*	*		CRF
Shrestha (2016)	*	*	*	*	CRF
Sikdar and Gambäck (2016)	*	*	*	*	CRF
Xia (2016)	*	*	*		CRF
Clematide and Makarov (2017)	*	*	*		CRF
Gómez-Adorno et al. (2017)			*		NB (cf. §6.5), SVM (cf. §6.8)
Martinc et al. (2017)				*	SVM, LR, RF (cf. §6.2), ...

Table 6: References (2016-) where prefixes and suffixes collected from a training corpus have been used for LI. The columns indicate the length of the prefixes and suffixes. The method column indicates the method used. The relevant section numbers are mentioned in parentheses. "..." indicates that even more methods were used.

et al., 2008).[5] You et al. (2008) used the trigrams of non-syllable chunks that were based on MI. Yeong and Tan (2010) experimented also with Markovian bigrams using both character and grapheme bigrams, but the syllable bigrams proved to work better. Graphemes in this case are the minimal units of the writing system, where a single character may be composed of several graphemes (e.g. in the case of the Hangul or Thai writing systems). Later, Yeong and Tan (2011) also used grapheme uni- and trigrams. Yeong and Tan (2011) achieved their best results combining word unigrams and syllable bigrams with a grapheme back-off.

Elfardy, Al-Badrashiny, and Diab (2014) used the MADAMIRA toolkit (Pasha et al., 2014) for D3 decliticization and then used D3-token 5-grams. D3 decliticization is a way to preprocess Arabic words presented by Habash and Sadat (2006). They evaluated the performance of an otherwise similar language identifier using character 5-grams, and found that the one using D3-token 5-grams performed somewhat better (F-score of 77.6 vs. 73.7) over the development set of the shared task at the EMNLP 2014 workshop on Computational Approaches to Code Switching (Solorio et al., 2014).

Graphones are sequences of characters linked to sequences of corresponding phonemes. They are automatically deduced from a bilingual corpus which consists of words and their correct pronunciations using Joint Sequence Models ("JSM") (Bisani & Ney, 2008). Giwa and Davel (2014) used language tags instead of phonemes when generating the graphones and then used Markovian graphone n-grams from 1 to 8 in LI. In word-level LI between four South African languages, their identifier obtained an accuracy of 97.2%.

5.5 Words

Position of words Kumar, Kumar, and Soman (2015) used the position of the current word in word-level LI. The position of words in sentences has also been used as a feature in code-switching point prediction by Dongen (2017).

5. They can be considered to be variable-length character n-grams that have been selected using certain criteria.

Word characteristics
Frequency of each possible character
Number of different characters
Number of different vowels
Whether the first and last characters were consonants or vowels
Number of character twins in two categories: vowel-vowel or consonant-consonant
Number of diphthongs
Overall number of syllables as well as the number of syllables with one, two or three characters
Number of characters in the first and last syllables
Number of syllables by type: AB, BA, ALB, BAI or BAL
(A, I stand for vowels and B, L stand for consonants)

Table 7: Word characteristics used by Mustonen (1965).

The characteristics of words Mustonen (1965) used the characteristics of words as parts of discriminating functions. The characteristics used by Mustonen (1965) are listed in Table 7. Barman, Wagner, Chrupala, and Foster (2014b) used string edit distance and n-gram overlap between the word to be identified and words in dictionaries. Similarly, Jhamtani et al. (2014) used a modified edit distance, which considers common spelling substitutions when Hindi is written using Latin characters. Das and Gambäck (2013) used the Minimum Edit Distance ("MED").

Basic dictionary Basic dictionaries are unordered lists of words belonging to a language. Basic dictionaries do not include information about word frequency, and are independent of the dictionaries of other languages. Vitale (1991) used a dictionary for LI as a part of his speech synthesizer. Each word in a dictionary had only one possible "language", or pronunciation category. More recently, a basic dictionary has been used for LI by Adouane and Dobnik (2017), Dongen (2017), and Duvenhage et al. (2017). The lexicon based identifier used by Adouane and Dobnik (2017) achieved an accuracy of 81.98% in word level LI when distinguishing between 6 languages and 3 additional categories.[6] The test texts were multilingual documents mainly written in Algerian Arabic; the accuracy for the majority-class baseline – that is every word identified as Algerian Arabic – was 55.10%.

Dictionary of unique words Unique word dictionaries include only those words of the language, that do not belong to the other languages targeted by the language identifier. Kulikowski (1991) used unique short words (from one to three characters) to differentiate between languages. Recently, a dictionary of unique words was used for LI by Adouane (2016), Guellil and Azouaou (2016), and Martinc et al. (2017). Language variety specific word lists were part of those features evaluated by Martinc et al. (2017) with FeatureUnion in scikit-learn; they had equal weight to the character flood counts mentioned in Section 5.3.

Specific classes of words Giguet (1995) used exhaustive lists of function words collected from dictionaries. Wechsler, Páraic, and Schäuble (1997) used stop words – that is non-content or closed-class words – as a training corpus. Similarly, Lins and Gonçalves (2004) used words from closed word classes, and Stupar et al. (2011) used lists of function words.

6. The languages and categories were: Algerian Arabic, modern standard Arabic, French, Berber, English, borrowings, named entities, and sounds and interjections.

Al-Badrashiny, Elfardy, and Diab (2015) used a lexicon of Arabic words and phrases that convey modality. Common to these features is that they are determined based on linguistic knowledge.

Discriminating words Rehůřek and Kolkus (2009) used the most relevant words for each language. Babu and Kumar (2010) used unique or nearly unique words. Franco-Salvador et al. (2015a) used Information Gain Word-Patterns ("IG-WP") to select the words with the highest information gain.

Most common words Souter, Churcher, Hayes, Hughes, and Johnson (1994) made an (unordered) list of the most common words for each language, as, more recently, did Cazamias, Dixit, and Marek (2015), Panich (2015), and Abainia et al. (2016). Pavan, Tandon, and Varma (2010) encoded the most common words to root forms with the Soundex algorithm.

Word frequency Mather (1998) collected the frequencies of words into feature vectors. Ács et al. (2015) used TF-IDF scores of words to distinguish between language groups. Recently, the frequency of words has also been used for LI by Clematide and Makarov (2017) and Gómez-Adorno et al. (2017). Gómez-Adorno et al. (2017) used the frequency of words and character n-grams as features in the DSL 2017 shared task (Zampieri et al., 2017). Their two-staged language identifier using SVM in language group identification and NB within groups achieved 6th position, with an accuracy of 91.46%.

The relative frequency of words Poutsma (2002) and Zhdanova (2002) were the first to use relative frequencies of words in LI. Jauhiainen (2010) found that combining the use of character n-grams with the use of words provided the best results. His language identifier obtained 99.8% average recall for 50 character sequences for the 10 evaluated languages (choosing between the 13 languages known by the language identifier) when using character n-grams from 1 to 6 combined with words. Tiedemann and Ljubešić (2012) calculated the relative frequency of words over all the languages. Artemenko and Shramko (2005) calculated the IDF of words, following the approach outlined in Equation 8. Xu et al. (2016) calculated the Pointwise Mutual Information ("PMI") for words and used it to group words to Chinese dialects or dialect groups. Recently, the relative frequency of words has also been used for LI by Jauhiainen et al. (2017a, 2017b) and Jourlin (2017)

Short words Grefenstette (1995) used the relative frequency of words with less than six characters. Recently, Panich (2015) also used short words, as did Simaki et al. (2017). The Relief feature selection algorithm used by Simaki et al. (2017) to rank the importance of different features placed short words (of less than three characters) at 8th position out of the 31 ranked features.

Search engine queries Alex (2005) used the relative frequency calculated from Google searches. Google was later also used by You et al. (2008) and Yang and Liang (2010).

Word probability maps Scherrer and Rambow (2010) created probability maps for words for German dialect identification between six dialects. In a word probability map, each predetermined geographic point has a probability for each word form. Probabilities were derived using a linguistic atlas and automatically-induced dialect lexicons.

Morphological analyzers and spellchecking Pienaar and Snyman (2010) used commercial spelling checkers, which utilized lexicons and morphological analyzers. The language identifier of Pienaar and Snyman (2010) obtained 97.9% accuracy when classifying one-line texts between 11 official South African languages. Elfardy and Diab (2012) used the AL-MORGEANA analyzer (Habash, 2007) to check if the word had an analysis in Modern Standard Arabic. They also used sound change rules to use possible phonological variants with the analyzer. Joshi, Bhatt, and Patel (2013) used spellchecking and morphological analyzers to detect English words from Hindi–English mixed search queries. Akosu and Selamat (2014) used spelling checkers to distinguish between 15 languages, extending the work of Pienaar and Snyman (2010) with dynamic model selection in order to gain better performance. Shrestha (2014) used a similarity count to find if mystery words were misspelled versions of words in a dictionary.

Word clusters Pham and Tran (2003) used an "LBG-VQ" (Linde, Buzo & Gray algorithm for Vector Quantization) approach to design a codebook for each language (Linde, Buzo, & Gray, 1980). The codebook contained a predetermined number of codevectors. Each codeword represented the word it was generated from as well as zero or more words close to it in the vector space.

5.6 Word Combinations

Sentence length Elfardy and Diab (2013) used the number of words in a sentence with NB. van der Lee and Bosch (2017) and Simaki et al. (2017) used the sentence length calculated in both words and characters with several machine learning algorithms. The feature selection algorithm used by Simaki et al. (2017) placed the sentence length calculated in characters in 14th position, or 29th position when the sentence length was calculated in words (out of the 31 highest-ranking features).

Statistics of words van der Lee and Bosch (2017) used the ratio to the total number of words of: once-occurring words, twice-occurring words, short words, long words, function words, adjectives and adverbs, personal pronouns, and question words. They also used the word-length distribution for words of 1–20 characters. Together with a variety of other features, their language identifier was able to distinguish between the Netherlandic and the Flemish Dutch varieties with 88% accuracy.

Article	1	2	3	4	5	6	7	Method
Bhattu and Ravi (2015)	***	***	***	***	***			LR (cf. §6.6)
Bobicev (2015)		***						PPM-C (cf. §5.7)
Ghosh, Ghosh, and Das (2015)	***	***	***	***	***	***	***	CRF (cf. §10.7)
Huang (2015)	***	*	*					Product (cf. §6.5)
Qiao and Lévy (2015)	*	*						SVM (cf. §6.8)
Raghavi, Chinnakotla, and Shrivastava (2015)	***	***	***	***	***	***	***	SVM
Shah et al. (2015)		***	***	***	***			SVM
Adouane et al. (2016c, 2016a)	***	**	**	*	*			SVM
Adouane (2016)	***	***	**	*				NB (cf. §6.5), SVM, LR, kNN (cf. §6.11), DT (cf. §6.2)
Barbaresi (2016)		***						RF (cf. §6.2)
Eldesouki et al. (2016)	***	*	*					SVM, LR, NN (cf. §6.9), NB
Franco-Penya and Sanchez (2016)	*	***						NB, SVM
Hanani et al. (2016)	***	***	***					LSTM RNN (cf. §6.9)
Samih and Maier (2016), Samih (2017)	***	***	***	***	***	***	***	CRF
Samih et al. (2016)	***	***	***	***	***			LSTM RNN - CRF
Schulz and Keller (2016)	***	***	***	***	***			CRF
Sikdar and Gambäck (2016)	***	***	***	***	***			CRF
Xia (2016)	***	***	***					CRF
Zampieri, Malmasi, Sulea, and Dinu (2016)	***	**						SVM
Alrifai et al. (2017)	*	*	*					SVM
Criscuolo and Aluísio (2017)	***	***						NN (cf. §6.9)
Gamallo et al. (2017)	***	***	***					Perplexity (cf. §6.6)
Hanani et al. (2017)	***	***	*					SVM
Kheng, Léa, and Granitzer (2017)	***	***	***					NB, SVM, RF
van der Lee and Bosch (2017)								AdaBoost (cf. §6.11), DT, SVM, NB, ...
Mathur et al. (2017)	**	***	*	*	*	*	*	NB, LR
Mendoza and Mendelsohn (2017)		***						Product
Miura et al. (2017)	***	***						SVM
Poulston, Waseem, and Stevenson (2017)	***	***						LR
Rangel et al. (2017a)		***						SVM, NB, NN, ...
Rijhwani, Sequeira, Choudhury, Bali, and Maddila (2017)	***	***	***					HMM (cf. §6.10)
Sanchez-Perez et al. (2017)	***	***						SVM
Tellez et al. (2017)	***	***	***					SVM

Table 8: References (2015–) where word n-grams have been used as features. The numbered columns indicate the length of the n-grams used. "***" indicates the best and "**" the second best n-gram length, as evaluated in the article in question. "*" indicates less effective n-gram lengths. Identical numbers of asterisks indicate that there was no clear order of effectiveness, or that all the n-gram lengths were used simultaneously. The method column indicates the method used. The relevant section numbers are mentioned in parentheses. "..." indicates that even more methods were used.

Word n-grams Marcadet et al. (2005) used at least the previous and following words with manual rules in word-level LI for text-to-speech synthesis. Rosner and Farrugia (2007) used Markovian word n-grams with a Hidden Markov Model ("HMM") tagger (Section 6.10). Table 8 lists more recent articles where word n-grams or similar constructs have been used. "PPM" in the methods column refers to Prediction by Partial Matching (Section 5.7), and "kNN" to k Nearest Neighbor classification (Section 6.11). In their final model, Mathur et al. (2017) ended up using only character n-grams and word unigrams, and did not use the word n-grams they experimented with in their preliminary experiments. Sanchez-Perez et al. (2017) evaluated different combinations of word and character n-grams with SVMs. The best combination for language variety identification was using all the features simultaneously. Tellez et al. (2017) used normal and gapped word n-grams and character n-grams simultaneously.

Singh (2006) used word trigrams simultaneously with character 4-grams. He concluded that word-based models can be used to augment the results from character n-grams when they are not providing reliable identification results. Table 9 lists articles where both character and word n-grams have been used together. "CBOW" in the methods column refer to Continuous Bag of Words neural network (Section 6.9), and "MIRA" to Margin Infused Relaxed Algorithm (Section 6.8).

Co-occurrences of words Wan (2016) uses word embeddings consisting of Positive Pointwise Mutual Information ("PPMI") counts to represent each word type. Then they use Truncated Singular Value Decomposition ("TSVD") to reduce the dimension of the word vectors to 100. Elgabou and Kazakov (2017) used k-means clustering when building dialectal Arabic corpora. Kheng et al. (2017) used features provided by Latent Semantic Analysis ("LSA") with SVMs and NB. LSA based features were used together with word n-grams in identifying Arabic and Spanish varieties with a SVM classifier as part of the language variety identification shared task at PAN 2017 (Rangel et al., 2017b). Their language identifier attained 16th position at the shared task.

Mikolov, Chen, Corrado, and Jeffrey (2013) present two models, the CBOW model and the continuous skip-gram model. The CBOW model can be used to generate a word given its context and the skip-gram model can generate the context given a word. The projection matrix, which is the weight matrix between the input layer and the hidden layer, can be divided into vectors, one vector for each word in the vocabulary. These word-vectors are also referred to as word embeddings. The embeddings can be used as features in other tasks after the neural network has been trained. Lin, Ammar, Levin, and Dyer (2014), Chang and Lin (2014), Franco-Salvador et al. (2015a), Franco-Salvador, Rosso, and Rangel (2015b), Jain (2015), Franco-Salvador, Kondrak, and Rosso (2017a), Franco-Salvador et al. (2017b), and Rangel et al. (2017a) used word embeddings generated by the word2vec skip-gram model (Mikolov et al., 2013) as features in LI. Poulston et al. (2017) used word2vec word embeddings and k-means clustering. Akhtyamova, Cardiff, and Ignatov (2017), Samih (2017), and Kodiyan, Hardegger, Neuhaus, and Cieliebak (2017) also used word embeddings created with word2vec. Kodiyan et al. (2017) used word embeddings with GRUs, and attained 12th place in language variety identification in the Twitter shared task at PAN 2017 (Rangel et al., 2017b).

Article	1	2	3	4	5	6	7	char	Method
Singh (2006, 2010)			✓					1-4	similarity measures (cf. §6.6)
Das and Gambäck (2013, 2014)	✓	✓	✓	✓	✓	✓	✓	1-7	SVM (cf. §6.8)
Nguyen and Dogruöz (2013)	✓	✓	✓					1-5	LR (cf. §6.6), CRF (cf. §10.7)
Chittaranjan, Vyas, Bali, and Choudhury (2014)	✓	✓	✓					1-5	CRF
Darwish et al. (2014)	✓	✓	✓					1-5	RF (cf. §6.2)
Goutte, Léger, and Carpuat (2014)	✓	✓						2-6	SVM, NB-like (cf. §6.5)
Gupta, Kumar, and Ekbal (2014)	✓	✓	✓	✓	✓			1-3	RF, SVM, DT (cf. §6.2), ...
King, Radev, and Abney (2014)	✓	✓						1-5	NB, LR, SVM
Ullman (2014)	✓	✓						1-4	NB-like
Ács et al. (2015)	✓	✓						1-4	LR, SVM
Chang and Lin (2014)	✓	✓	✓					2-3	RNN (cf. §6.9)
Goutte and Léger (2015)	✓	✓						2-6	SVM, NB-like
Jain (2015)	✓	✓	✓	✓	✓	✓	✓	2-4	CRF
Malmasi and Dras (2015a)	✓	✓						1-3	SVM
Malmasi and Dras (2015b)	✓	✓						1-6	SVM
Malmasi et al. (2015)	✓	✓						1-4	SVM
Castro et al. (2016)	✓	✓						2-7	Product (cf. §6.5)
Zirikly, Desmet, and Diab (2016)	✓	✓	✓					1-6	LR
Basile et al. (2017)	✓	✓						3-6	SVM
Castro et al. (2017)	✓	✓						2-7	NB, LR, SVM, RF
Ciobanu, Zampieri, Malmasi, and Dinu (2017)	✓	✓						1-6	SVM
Çöltekin and Rama (2017)	✓	✓	✓					1-7+	SVM
Markov, Gómez-Adorno, and Sidorov (2017)	✓	✓	✓					3-7	SVM, NB
Martinc et al. (2017)	✓	✓						4	SVM, LR, RF, ...
Medvedeva, Kroon, and Plank (2017)	✓	✓	✓	✓				1-6	SVM, CBOW (cf. §6.9)
Mendoza and Mendelsohn (2017)	✓	✓						1-6	SVM
Pla and Hurtado (2017)	✓	✓	✓	✓				1-6	SVM
Williams and Dagli (2017)	✓	✓	✓	✓	✓			1-5	MIRA (cf. §6.8)

Table 9: List of articles where word and character n-grams have been used as features. The numbered columns indicate the length of the word n-grams and char-column the length of character n-grams used. The method column indicates the method used. The relevant section numbers are mentioned in parentheses. "..." indicates that even more methods were used.

Çöltekin and Rama (2016) trained both character and word embeddings using the FastText text classification method (Joulin et al., 2017) on the DSL 2016 shared task, where it reached low accuracy when compared with the other methods. Xia (2016) used FastText to train word vectors including subword information. Then he used these word vectors together with some additional word features to train a CRF-model which was used for code-switching detection.

Barman et al. (2014b) extracted features from the hidden layer of a Recurrent Neural Network ("RNN") that had been trained to predict the next character in a string. They evaluated several features with a SVM classifier, and found the RNN-extracted features alone were far inferior to character n-grams. However, the RNN features slightly improved the results when they were added as additional information to the classifier on top of the other features.

Syntax and part-of-speech ("POS")-tags Alex (2005) evaluated methods for detecting foreign language inclusions and experimented with a Conditional Markov Model ("CMM") tagger, which had performed well on Named Entity Recognition ("NER") (Klein, Smarr, Nguyen, & Manning, 2003). Alex (2005) was able to produce the best results by incorporating her own English inclusion classifier's decision as a feature for the tagger, and not using the tagger's POS-tags. Romsdorfer and Pfister (2007) used syntactic parsers together with dictionaries and morpheme lexicons. Lui and Cook (2013) used n-grams composed of POS-tags and function words. Piergallini et al. (2016a) used labels from a NER system, cluster prefixes, and Brown clusters (Brown, deSouza, Mercer, Della Pietra, & Lai, 1992).

Adouane and Dobnik (2017) used POS-tag n-grams from one to three and Bestgen (2017) from one to five, and Martinc et al. (2017) used POS-tag trigrams with TF-IDF weighting. POS-tag trigrams were part of the features evaluated by Martinc et al. (2017) for Spanish variety identification, where they were found to have less importance than character repetition and lists of unique words (as mentioned earlier). Schulz and Keller (2016), Basile et al. (2017), van der Lee and Bosch (2017), and Simaki et al. (2017) also recently used POS-tags. Schulz and Keller (2016) used POS-tags together with character n-gram prefixes and suffixes in word level language identification in Latin–Middle English mixed text. Their CRF based classifier was able to attain a macro-averaged F-score of 67.4.

Franco-Salvador et al. (2015a) used POS-tags with emotion-labeled graphs in Spanish variety identification. In emotion-labeled graphs, each POS-tag was connected to one or more emotion nodes if a relationship between the original word and the emotion was found in the Spanish Emotion Lexicon. They also used POS-tags with IG-WP. Elfardy et al. (2014) used the MADAMIRA tool for morphological analysis disambiguation. The polySVOX text analysis module described by Romsdorfer and Pfister (2007) uses two-level rules and morpheme lexicons on the sub-word level and separate definite clause grammars (DCGs) on word, sentence, and paragraph levels. The language of sub-word units, words, sentences, and paragraphs in multilingual documents is identified at the same time as performing syntactic analysis of the document. Noh, Talib, Ahmad, Halim, and Mohamed (2009) converted sentences into POS-tag patterns using a word-POS dictionary for Malay. The POS-tag patterns were then used by a neural network to indicate whether the sentences were written in Malay or not. Laboreiro, Bošnjak, Sarmento, Rodrigues, and Oliveira (2013) used Jspell to detect differences in the grammar of Portuguese variants. Bekavac et al. (2014) used a syntactic grammar to recognize verb-*da*-verb constructions, which are characteristic of the Serbian language. The syntactic grammar was used together with several morphological grammars to distinguish between Croatian and Serbian.

Languages identified for surrounding words in word-level LI Marcadet et al. (2005) used the weighted LI scores of the words to the left and right of the word to be classified. Rosner and Farrugia (2007) used language labels within an HMM. Akhil and Ab-

hishek (2014) used the language labels of other words in the same sentence to determine the language of the ambiguous word. The languages of the other words had been determined by the positive Decision Rules (Section 6.1), using dictionaries of unique words when possible. Elfardy and Diab (2013) used the word-level language labels obtained with the approach of Elfardy, Al-Badrashiny, and Diab (2013) on sentence-level dialect identification. Das and Gambäck (2013, 2014) used the language tags of the previous three words with an SVM. Mukherjee, Ravi, and Datta (2014) used language labels of surrounding words with NB. King et al. (2015) used the language probabilities of the previous word to determining weights for languages. King et al. (2014) used unigram, bigram and trigram language label transition probabilities. Papalexakis, Nguyen, and Dogruöz (2014) used the language labels for the two previous words as well as knowledge of whether code-switching had already been detected or not. Raj and Karfa (2014) used the language label of the previous word to determine the language of an ambiguous word. Sinha and Srinivasa (2014) also used the language label of the previous word. Chanda, Das, and Mazumdar (2016a) used the language identifications of 2–4 surrounding words for post-identification correction in word-level LI. Samih and Maier (2016) used language labels, among other features, with a CRF when identifying code-switching between Modern Standard Arabic and Moroccan Darija Dialectal Arabic. With the best settings, their system was able to attain an accuracy of 91.4% at the word level. Dongen (2017) used language labels of the current and two previous words in code-switching point prediction. The predictive strength of the current label was lower than the count of code-switches, but better than the length or position of the word. All of the features were used together with NB, DT and SVM. Guzmán, Ricard, Serigos, Bullock, and Toribio (2017) used language label bigrams with an HMM together with a 5-gram character model in word-level language identification of texts including Spanish–English and French–English code-switching. They achieved word-level accuracy rates of 95% and 97%, respectively.

5.7 Feature Smoothing

Feature smoothing is required in order to handle the cases where not all features f_i in a test document have been attested in the training corpus for a certain language. Thus, it is used especially when the count of features is high, or when the amount of training data is low. Smoothing is usually handled as part of the method, and not pre-calculated into the language models. Most of the smoothing methods evaluated by Chen and Goodman (1999) have been used in LI, and we follow the order of methods in that article.

Additive smoothing (Laplace, Lidstone) In Laplace smoothing, an extra number of occurrences is added to every possible feature in the language model. Dunning (1994) used Laplace's sample size correction (add-one smoothing) with the product of Markovian probabilities. Adams and Resnik (1997) experimented with additive smoothing of 0.5, and noted that it was almost as good as Good-Turing smoothing. Chen and Goodman (1999) calculate the values for each n-gram as:

$$v_{C_g}(f) = \frac{c(C_g, f) + \lambda}{l_{C_g^n} + |U(C_g^n)|\lambda} \tag{11}$$

where $v_{C_g}(f)$ is the probability estimate of n-gram f in the model and $c(C_g, f)$ its frequency in the training corpus. $l_{C_g^n}$ is the total number of n-grams of length n and $|U(C_g^n)|$ the number of distinct n-grams in the training corpus. λ is the Lidstone smoothing parameter. When using Laplace smoothing, λ is equal to 1 and with Lidstone smoothing, the λ is usually set to a value between 0 and 1.

The penalty values used by Jauhiainen et al. (2016) with the HeLI method function as a form of additive smoothing. Vatanen et al. (2010) evaluated additive, Katz, absolute discounting (Ney, Essen, & Kneser, 1994), and Kneser-Ney smoothing methods. Additive smoothing produced the least accurate results of the four methods. Cann (2015) and Franco-Penya and Sanchez (2016) evaluated NB with several different Lidstone smoothing values. Cianflone and Kosseim (2016) used additive smoothing with character n-grams as a baseline classifier, which they were unable to beat with Convolutional Neural Networks ("CNNs").

Good-Turing Discounting Adams and Resnik (1997) used Good-Turing smoothing with the product of Markovian probabilities. Chen and Goodman (1999) define the Good-Turing smoothed count $c_{GT}(C, f)$ as:

$$c_{GT}(C_g, f) = (c(C_g, f) + 1)\frac{r_{c(C_g,f)+1}}{r_{c(C_g,f)}} \tag{12}$$

where $r_{c(C_g,f)}$ is the number of features occurring exactly $c(C_g, f)$ times in the corpus C_g. More recently, Good-Turing smoothing has been used by Giwa (2016) and Gamallo, Pichel, Alegria, and Agirrezabal (2016). Gamallo et al. (2016) used the smoothing for unseen words with a NB classifier, attaining 13th place in DSL 2016 shared task (Malmasi et al., 2016).

Jelinek-Mercer Rehůřek and Kolkus (2009) used Jelinek-Mercer smoothing correction over the relative frequencies of words, calculated as follows:

$$v_{C_g}(f) = \lambda\frac{c(C, f)}{l_{C^F}} + (1 - \lambda)\frac{c(C_g, f)}{l_{C_g^F}} \tag{13}$$

where λ is a smoothing parameter, which is usually some small value like 0.1. Mendizabal et al. (2014) used character 1–8 grams with Jelinek-Mercer smoothing. Their language identifier using character 5-grams achieved 3rd place (out of 12) in the TweetLID shared task constrained track (Zubiaga et al., 2016).

Katz Ramisch (2008) and Vatanen et al. (2010) used the Katz back-off smoothing (Katz, 1987) from the SRILM toolkit (Stolcke, 2002), with perplexity. Katz smoothing is an extension of Good-Turing discounting. The probability mass left over from the discounted n-grams is then distributed over unseen n-grams via a smoothing factor. In the smoothing evaluations by Vatanen et al. (2010), Katz smoothing performed almost as well as absolute discounting, which produced the best results. Giwa and Davel (2013) evaluated Witten-Bell, Katz, and absolute discounting smoothing methods. Witten-Bell got 87.7%, Katz 87.5%, and absolute discounting 87.4% accuracy with character 4-grams.

Prediction by Partial Matching (PPM/Witten-Bell) Teahan (2000) used the PPM-C algorithm for LI. PPM-C is basically a product of Markovian probabilities with an escape scheme. If an unseen context is encountered for the character being processed, the escape

probability is used together with a lower-order model probability. In PPM-C, the escape probability is the sum of the seen contexts in the language model. PPM-C was lately used by Adouane et al. (2016d). The PPM-D+ algorithm was used by Celikel (2005). Bergsma, McNamee, Bagdouri, Fink, and Wilson (2012) and McNamee (2016) used a PPM-A variant. Yamaguchi and Tanaka-Ishii (2012) also used PPM. The language identifier of Yamaguchi and Tanaka-Ishii (2012) obtained 91.4% accuracy when classifying 100 character texts between 277 languages. Jaech, Mulcaire, Hathi, Ostendorf, and Smith (2016b) used Witten-Bell smoothing with perplexity.

Herman, Suchomel, Baisa, and Rychlý (2016) used a Chunk-Based Language Model ("CBLM"), which is similar to PPM models.

Absolute discounting Vatanen et al. (2010) used several smoothing techniques with Markovian probabilities. Absolute discounting from the VariKN toolkit performed the best. Vatanen et al. (2010) define the smoothing as follows: a constant D is subtracted from the counts $c(C_g, u_{i-n+1,\ldots,i})$ of all observed n-grams $u_{i-n+1,\ldots,i}$ and the held-out probability mass is distributed between the unseen n-grams in relation to the probabilities of lower order n-grams $P_g(u_i|u_{i-n+2,\ldots,i-1})$, as follows:

$$P_{C_g}(u_i|u_{i-n+1,\ldots,i-1}) = \frac{c(C_g, u_{i-n+1,\ldots,i}) - D}{c(C_g, u_{i-n+1,\ldots,i-1})} + \lambda_{u_{i-n+1,\ldots,i-1}} P_{C_g}(u_i|u_{i-n+2,\ldots,i-1}) \quad (14)$$

where $\lambda_{u_{i-n+1,\ldots,i-1}}$ is a scaling factor that makes the conditional distribution sum to one. Absolute discounting with Markovian probabilities from the VariKN toolkit was later also used by Rodrigues (2012), Maier and Gómez-Rodríguez (2014), and Jauhiainen et al. (2017b).

Kneser-Ney smoothing The original Kneser-Ney smoothing is based on absolute discounting with an added back-off function to lower-order models (Vatanen et al., 2010). Chen and Goodman (1999) introduced a modified version of the Kneser-Ney smoothing using interpolation instead of back-off. Chen and Maison (2003) used the Markovian probabilities with Witten-Bell and modified Kneser-Ney smoothing. Giwa (2016), Balažević et al. (2016), and Rijhwani et al. (2017) also recently used modified Kneser-Ney discounting. Barbaresi (2016) used both original and modified Kneser-Ney smoothings. In the evaluations of Vatanen et al. (2010), Kneser-Ney smoothing fared better than additive, but somewhat worse than the Katz and absolute discounting smoothing. Lately Samih and Maier (2016) also used Kneser-Ney smoothing.

Castro et al. (2016, 2017) evaluated several smoothing techniques with character and word n-grams: Laplace/Lidstone, Witten-Bell, Good-Turing, and Kneser-Ney. In their evaluations, additive smoothing with 0.1 provided the best results. Good-Turing was not as good as additive smoothing, but better than Witten-Bell and Kneser-Ney smoothing. Witten-Bell proved to be clearly better than Kneser-Ney.

6. Methods

In recent years there has been a tendency towards attempting to combine several different types of features into one classifier or classifier ensemble. Many recent studies use readily available classifier implementations and simply report how well they worked with the feature

set used in the context of their study. There are many methods presented in this article that are still not available as out of the box implementations, however. There are many studies which have not been re-evaluated at all, going as far back as Mustonen (1965). In this survey, we consider two methods to be the same if they always produce exactly the same results from exactly the same inputs. Our hope is that this article will inspire new studies and many previously unseen ways of combining features and methods. In the following sections, the reviewed articles are grouped by the methods used for LI.

Rubinstein and Hastie (1997) divide classification methods into informative and discriminative classifiers. *Generative*[7] classifiers aim to model the underlying phenomenon, and classification is performed by calculating a probability for the observations using the model of each class. *Discriminative* classifiers do not try to model the phenomenon itself, but are instead modeling the class boundaries or the class probabilities directly. As examples of generative classifiers, Rubinstein and Hastie (1997) list Fisher Discriminant Analysis, Hidden Markov Models (Section 6.10), and Naive Bayes (Section 6.5), and as examples of discriminative classifiers, Logistic Regression (Section 6.6), Neural Networks (Section 6.9), and Generalized Additive Models.

Ng and Jordan (2002) use Logistic Regression as an example of a discriminative classifier and Naive Bayes as an example of a generative classifier. Experimenting with the two methods, they show that if the amount of training material is large enough, discriminative classifiers usually attain better results. However, in many cases the generative classifier obtains better results when the amount of training data is small.

In generative probabilistic modeling, it is possible to calculate the probability of a given text using language models, irrespective of the other languages being considered. Adding new languages to generative systems only requires modeling the languages to be added, but in a system using a discriminative classifier, all the languages have to be re-trained. With discriminative models, new borders for the remaining languages also have to be learned when removing languages from the repertoire. Some commonly-used techniques, such as tuning hyperparameters using a development corpus, can move otherwise generative classifiers towards the discriminative side.

6.1 Decision Rules

Henrich (1989) used positive Decision Rules with unique characters and character n-grams, that is, if a unique character or character n-gram was found, the language was identified. The positive Decision Rule (unique features) for the test document M and the training corpus C_g can be formulated as follows:

$$R_{DR^+}(g, M) = \begin{cases} 1 & \text{, if } \exists f \in U(M) : c(C_g, f) > 0 \wedge c(C_j, u) = 0 \wedge g \neq j \\ 0 & \text{, otherwise} \end{cases} \quad (15)$$

where $U(M)$ is the set of unique features in M, C_g is the corpus for language g, and C_j is a corpus of any other language j. Positive decision rules can also be used with non-unique features when the decisions are made in a certain order. For example, Dongen (2017) presents the pseudo code for her dictionary lookup tool, where these kind of decisions are part of an if-then-else statement block. Her (manual) rule-based dictionary lookup

7. We follow (Ng & Jordan, 2002) and refer to informative classifiers as generative classifiers.

tool works better for Dutch–English code-switching detection than the SVM, DT, or CRF methods she experiments with. The positive Decision Rule has also been used recently by Abainia et al. (2016), Chanda et al. (2016b, 2016a), Guellil and Azouaou (2016), Gupta, Bhatt, and Mittal (2016), Adouane and Dobnik (2017), and He et al. (2016). He et al. (2016) used the positive decision rule with unique characters in distinguishing between Uyghur and Kazakh. If the text did not include any unique characters then the language of the text was identified using other features with a LR classifier.

In the negative Decision Rule, if a character or character combination that was found in M does not exist in a particular language, that language is omitted from further identification. The negative Decision Rule can be expressed as:

$$R_{DR^-}(g, M) = \begin{cases} 0 & \text{, if } \exists f \in U(M) : c(C_g, f) = 0 \\ 1 & \text{, otherwise} \end{cases} \tag{16}$$

where C_g is the corpus for language g. The negative Decision Rule was first used by Giguet (1995) in LI.

Alshutayri et al. (2016) evaluated the JRIP classifier from the Waikato Environment for Knowledge Analysis ("WEKA") (Hall et al., 2009). JRIP is an implementation of the propositional rule learner (Cohen, 1995). It was found to be inferior to the SVM, NB and DT algorithms.

In isolation the desicion rules tend not to scale well to larger numbers of languages (or very short test documents), and are thus mostly used in combination with other LI methods or as a Decision Tree.

6.2 Decision Trees

Häkkinen and Tian (2001) were the earliest users of Decision Trees ("DTs") in LI. They used DTs based on characters and their context, without any frequency information. In training the DT, each node is split into child nodes according to an information theoretic optimization criterion. For each node, a feature is chosen which maximizes the information gain at that node. The information gain is calculated for each feature, and the feature with the highest gain is selected for the node. In the identification phase, the nodes are traversed until only one language is left (i.e. a leaf node is reached). Later, Ceylan and Kim (2009), Eskander, Al-Badrashiny, Habash, and Rambow (2014), and Moodley (2016) have been especially successful in using DTs. Moodley (2016) used DTs with character trigrams and was able to achieve an F-score of 68.76 at the word level when distinguishing between 11 languages used in South Africa.

A Random Forest (RF) is an ensemble classifier generating many DTs (Breiman, 2001). It has been succesfully used in LI by Jhamtani et al. (2014), Darwish et al. (2014), Ranjan, Raja, Priyadharshini, and Balabantaray (2016), and Malmasi and Zampieri (2017a, 2017b). Malmasi and Zampieri (2017a, 2017b) used RFs as a meta-classification algorithm in an SVM ensemble. As an ensemble method, the RF-classifier outperformed both voting and mean probability ensembles, resulting in first place on the German Dialect Identification ("GDI") shared task and second place on the ADI shared tasks at the VarDial Evaluation Campaign 2017 (Zampieri et al., 2017).

6.3 Simple Scoring

In simple scoring, each feature in the test document is checked against the language model for each language, and languages which contain that feature are given a point, as follows:

$$R_{simple}(g, M) = \sum_{i=1}^{l_{MF}} \begin{cases} 1 & \text{, if } f_i \in dom(O(C_g)) \\ 0 & \text{, otherwise} \end{cases} \tag{17}$$

where f_i is the ith feature found in the test document M. The language scoring the most points is the winner. Simple scoring is still a good alternative when facing an easy problem such as preliminary language group identification. It was recently used for this purpose by Franco-Salvador et al. (2015b) with a basic dictionary. They achieved 99.8% accuracy when distinguishing between 6 language groups. Kadri and Moussaoui (2013) use a version of simple scoring as a distance measure, assigning a penalty value to features not found in a model. In this version, the language scoring the least amount of points is the winner. Their language identifier obtained a 100% success rate with character 4-grams when classifying relatively large documents (from 1 to 3 kilobytes), between 10 languages. Simple scoring was also used lately by Balažević et al. (2016), Selamat and Akosu (2016), and Duvenhage et al. (2017).

6.4 Sum or Average of Values

The sum of values can be expressed as:

$$R_{sum}(g, M) = \sum_{i=1}^{l_{MF}} v_{C_g}(f_i) \tag{18}$$

where f_i is the ith feature found in the test document M, and $v_{C_g}(f_i)$ is the value for the feature in the language model of the language g. The language with the highest score is the winner.

The simplest case of Equation 18 is when the text to be identified contains only one feature. An example of this is Shrestha (2014) who used the frequencies of short words as values in word-level identification. For longer words, he summed up the frequencies of different-sized n-grams found in the word to be identified. Giwa and Davel (2014) first calculated the language corresponding to each graphone. They then summed up the predicted languages, and the language scoring the highest was the winner. When a tie occurred, they used the product of the Markovian graphone n-grams. Their method managed to outperform SVMs in their tests.

Henrich (1989) used the average of all the relative frequencies of the n-grams in the text to be identified as in Equation 19. Vogel and Tresner-Kirsch (2012) evaluated several variations of the LIGA algorithm introduced by Tromp and Pechenizkiy (2011). LIGA is basically the sum of relative frequencies using character tri- and 4-grams. Moodley (2016) and Jauhiainen et al. (2017b) also used LIGA and logLIGA methods. In the experiments of Jauhiainen et al. (2017b), LIGA and logLIGA were clearly inferior to the other evaluated methods. The average or sum of relative frequencies was also used recently by Abainia et al. (2016) and Martadinata et al. (2016). Martadinata et al. (2016) compared the sum of

the relative frequencies of words to the out-of-place method (Section 6.6) and the product of Markovian probabilities (Section 6.5). The sum of relative frequencies was not able to attain the same level of accuracy as the two other methods when distinguishing between four Indonesian languages.

Ng and Selamat (2009) summed up LFDF values (see Section 5.2), obtaining 99.75% accuracy when classifying document sized texts between four languages using Arabic script. Vitale (1991) calculates the score of the language for the test document M as the average of the probability estimates of the features, as follows:

$$R_{avg}(g, M) = \sum_{i=1}^{l_{M^F}} \frac{v_{C_g}(f_i)}{l_{M^F}} \tag{19}$$

where l_{M^F} is the number of features in the test document M. Brown (2013) summed weighted relative frequencies of character n-grams, and normalized the score by dividing by the length (in characters) of the test document. Normalizing the sums by document length does not change the order of the scored languages, but it gives comparable results between different lengths of test documents.

Vega and Bressan (2001a, 2001b) summed up the feature weights and divided them by the number of words in the test document in order to set a threshold to detect unknown languages. Their language identifier obtained 89% precision and 94% recall when classifying documents between five languages. El-Shishiny et al. (2004) used a weighting method combining alphabets, prefixes, suffixes and words. Elfardy and Diab (2012) summed up values from a word trigram ranking, basic dictionary and morphological analyzer lookup. Akhil and Abhishek (2014) summed up language labels of the surrounding words to identify the language of the current word. Bekavac et al. (2014) summed up points awarded by the presence of morphological and syntactic features. Gamallo et al. (2014) used inverse rank positions as values.

Ács et al. (2015) computed the sum of keywords weighted with TF-IDF to distinguish between language groups in the 2015 DSL shared task (Zampieri et al., 2015). Their language group classifier achieved 100% accuracy. Fabra-Boluda, Rangel, and Rosso (2015) summed up the TF-IDF derived probabilities of words. They used the sum as one of six features given to a Bayesian Net classifier.

6.5 Product of Values

The product of values can be expressed as follows:

$$R_{prod}(g, M) = \prod_{i}^{l_{M^F}} v_{C_g}(f_i) \tag{20}$$

where f_i is the ith feature found in test document M, and $v_{C_g}(f_i)$ is the value for the feature in the language model of language g. The language with the highest score is the winner. Some form of feature smoothing is usually required with the product of values method to avoid multiplying by zero.

Product of relative frequencies Church (1985) was the first to use the product of relative frequencies and it has been widely used ever since; recent examples include Castro et al. (2016, 2017), Hanani et al. (2017), and Jauhiainen et al. (2017b). Castro et al. (2016, 2017) used the product of relative frequencies in discriminating between Brazilian and European Portuguese in tweets. As mentioned earlier, they evaluated several smoothing techniques and the best configuration was an ensemble combining Good-Turing smoothed word unigrams, Witten-Bell smoothed word bigrams, and Lidstone smoothed character 6-grams. The ensemble achieved an average F1-score of 0.9272 for tweets.

Some of the authors use a sum of log frequencies rather than a product of frequencies to avoid underflow issues over large numbers of features, but the two methods yield the same relative ordering, with the proviso that the maximum of multiplying numbers between 0 and 1 becomes the minimum of summing their negative logarithms, as can be inferred from:

$$R_{logsum}(g, M) = -\log(R_{prod}(g, M)) = -\log \prod_{i=1}^{l_{MF}} v_{C_g}(f_i) = \sum_{i=1}^{l_{MF}} -\log(v_{C_g}(f_i)) \tag{21}$$

Naive Bayes (NB) When (multinomial[8]) NB is used in LI, each feature used has a probability to indicate each language. The probabilities of all features found in the test document are multiplied for each language, and the language with the highest probability is selected, as in Equation 20. Theoretically the features are assumed to be independent of each other, but in practice using features that are functionally dependent can improve classification accuracy (Peng & Schuurmans, 2003).

NB implementations have been widely used for LI, usually with a more varied set of features than simple character or word n-grams of the same type and length. The features are typically represented as feature vectors given to a NB classifier. Mukherjee et al. (2014) trained a NB classifier with language labels of surrounding words to help predict the language of ambiguous words first identified using an SVM. The language identifier used by Tan et al. (2014) obtained 99.97% accuracy with 5-grams of characters when classifying sentence-sized texts between six language groups. Goutte et al. (2014) used a probabilistic model similar to NB. Bhattu and Ravi (2015) used NB and naive Bayes EM, which uses the Expectation–Maximization ("EM") algorithm in a semi-supervised setting to improve accuracy. Ljubešić and Kranjcić (2014) used Gaussian naive Bayes ("GNB", i.e. NB with Gaussian estimation over continuous variables) from scikit-learn.

Bayesian Network Classifiers In contrast to NB, in Bayesian networks the features are not assumed to be independent of each other. The network learns the dependencies between features in a training phase. Fabra-Boluda et al. (2015) used a Bayesian Net classifier in two-staged (group first) LI over the open track of the DSL 2015 shared task coming second out of the three teams that participated. Rangel et al. (2017a) similarly evaluated Bayesian Nets, but found them to perform worse than the other 11 algorithms they tested.

Product of Markovian probabilities House and Neuburg (1977) used the product of the Markovian probabilities of character bigrams. The language identifier created by Brown (2013, 2014b), "whatlang", obtains 99.2% classification accuracy with smoothing

8. To the best of our knowledge, the multivariate Bernoulli version of NB has never been used for LI. See Giwa (2016) for a possible explanation.

for 65 character test strings, when distinguishing between 1,100 languages. The product of Markovian probabilities has recently also been used by Samih and Maier (2016) and Mendoza and Mendelsohn (2017). Mendoza and Mendelsohn (2017) evaluated several machine learning algorithms using the DSL 2016 dataset, and found the product of Markovian probabilities with word bigrams to perform clearly worse than the other evaluated methods: Logistic Regression (Section 6.6), Random Forest (Section 6.2), and several SVM configurations (Section 6.8).

HeLI Jauhiainen et al. (2016) use a word-based backoff method called HeLI. Here, each language is represented by several different language models, only one of which is used for each word found in the test document. The language models for each language are: a word-level language model, and one or more models based on character n-grams of order 1 to n_{max}. When a word that is not included in the word-level model is encountered in a test document, the method backs off to using character n-grams of the size n_{max}. If there is not even a partial coverage here, the method backs off to lower order n-grams and continues backing off until at least a partial coverage is obtained (potentially all the way to character unigrams). The LI system of Jauhiainen et al. (2016) implementing the HeLI method attained shared first place in the closed track of the DSL 2016 shared task (Malmasi et al., 2016), and was the best method tested by Jauhiainen et al. (2017b) for test documents longer than 30 characters.

6.6 Similarity Measures

Out-of-place method The well-known method of Cavnar and Trenkle (1994) uses overlapping character n-grams of varying sizes based on words. The language models are created by tokenizing the training texts for each language g into words, and then padding each word with spaces, one before and four after. Each padded word is then divided into overlapping character n-grams of sizes 1–5, and the counts of every unique n-gram are calculated over the training corpus. The n-grams are ordered by frequency and k of the most frequent n-grams, $f_1, ..., f_k$, are used as the domain of the language model $O(C_g)$ for the language g. The rank of an n-gram f in language g is determined by the n-gram frequency in the training corpus C_g and denoted $\text{rank}_{C_g}(f)$.

During LI, the test document M is treated in a similar way and a corresponding model $O(M)$ of the K most frequent n-grams is created. Then a distance score is calculated between the model of the test document and each of the language models. The value $v_{C_g}(f)$ is calculated as the difference in ranks between $\text{rank}_{C_g}(f)$ and $\text{rank}_M(f)$ of the n-gram f in the domain $dom(O(M))$ of the model of the test document. If an n-gram is not found in a language model, a special penalty value p is added to the total score of the language for each missing n-gram. The penalty value should be higher than the maximum possible distance between ranks.

$$v_{C_g}(f) = \begin{cases} |\,\text{rank}_M(f) - \text{rank}_{C_g}(f)| & \text{, if } f \in dom(O(C_g)) \\ p & \text{, if } f \notin dom(O(C_g)) \end{cases} \tag{22}$$

The score $R_{CT}(g)$ for each language g is the sum of values, as in Equation 18. The language with the lowest score $R_{CT}(g)$ is selected as the identified language. The method is equivalent to Spearman's measure of disarray (Diaconis & Graham, 1977). The out-of-place method

has been widely used in LI literature as a baseline. In the evaluations of Jauhiainen et al. (2017b) for 285 languages, the out-of-place method achieved an F-score of 95% for 35-character test documents. It was the fourth best of the seven evaluated methods for test document lengths over 20 characters.

Local Rank Distance ("LRD") Local Rank Distance (Ionescu, 2013) is a measure of difference between two strings. LRD is calculated by adding together the distances separating comparable units (for example character n-grams) across two strings. The distance is only calculated within a local window of predetermined length. Ionescu and Popescu (2016) and Ionescu and Butnaru (2017) used LRD with a Radial Basis Function ("RBF") kernel (see Section 6.8). For learning, they experimented with both Kernel Discriminant Analysis ("KDA") and Kernel Ridge Regression ("KRR"). Ionescu and Butnaru (2017) won the ADI 2017 shared task with the LRD and KRR combination, and came 5th in the GDI 2017 shared task with the same system (Zampieri et al., 2017).

Franco-Salvador et al. (2017a) used KDA in language variety identification. They evaluated the use of string kernels with KDA against word embeddings with SVM and LR classifiers, and found the KDA classifier to perform the best. However, a linear interpolation combining the KDA and LR classifiers gained even higher results for all three datasets used.

Levenshtein distance Pavan et al. (2010) calculated the Levenshtein distance between the language models and each word in the test text. The similary score for each language was the inverse of the sum of the Levenshtein distances. Their language identifier obtained 97.7% precision when classifying texts from two to four words between five languages. Later Guellil and Azouaou (2016) used Levenshtein distance for Algerian dialect identification and Gupta et al. (2016) for query word identification. Gupta et al. (2016) used a version of the n-gram Markov model to choose the most relevant result from a list of words generated by the Levenshtein algorithm.

Probability difference Botha, Zimu, and Barnard (2007), Botha and Barnard (2007), Botha (2008), and Botha and Barnard (2012) calculated the difference between probabilities as in Equation 23.

$$R_{diff}(g, M) = \sum_i (v_M(f_i) - v_{C_g}(f_i)) \tag{23}$$

where $v_M(f_i)$ is the probability for the feature f_i in the test text and $v_{C_g}(f_i)$ the corresponding probability in the language model of the language g. The language with the lowest score $R(g)$ is selected as the most likely language for the test text. Singh (2006, 2010) used the log probability difference and the absolute log probability difference. The log probability difference proved slightly better, obtaining a precision of 94.31% using both character and word n-grams when classifying 100 character texts between 53 language-encoding pairs.

Vectors Depending on the algorithm, it can be easier to view language models as vectors of weights over the target features. In the following methods, each language is represented by one or more feature vectors. Methods where each feature type is represented by only one feature vector are also sometimes referred to as centroid-based (Takçı & Güngör, 2012) or

nearest prototype methods. Distance measures are generally applied to all features included in the feature vectors.

Kruengkrai et al. (2005) calculated the squared Euclidean distance between feature vectors. The Squared Euclidean distance can be calculated as:

$$R_{\text{euc}^2}(g, M) = \sum_i (v_M(f_i) - v_{C_g}(f_i))^2 \tag{24}$$

Hamzah (2010) used the simQ similarity measure, which is closely related to the Squared Euclidean distance.

Stensby et al. (2010) investigated the LI of multilingual documents using a Stochastic Learning Weak Estimator ("SLWE") method. In SLWE, the document is processed one word at a time and the language of each word is identified using a feature vector representing the current word as well as the words processed so far. This feature vector includes all possible units from the language models – in their case mixed-order character n-grams from one to four. The vector is updated using the SLWE updating scheme to increase the probabilities of units found in the current word. The probabilities of units that have been found in previous words, but not in the current one, are on the other hand decreased. After processing each word, the distance of the feature vector to the probability distribution of each language is calculated, and the best-matching language is chosen as the language of the current word. Their language identifier obtained 96.0% accuracy when classifying sentences with ten words between three languages. They used the Euclidean distance as the distance measure as follows:

$$R_{\text{euc}}(g, M) = \sqrt{R_{\text{euc}^2}(g, M)} \tag{25}$$

Tomović and Janičić (2007) compared the use of Euclidean distance with their own similarity functions. Prager (1999) calculated the cosine angle between the feature vector of the test document and the feature vectors acting as language models. This is also called the cosine similarity and is calculated as follows:

$$R_{\cos}(g, M) = \frac{\sum_i v_M(f_i) v_{C_g}(f_i)}{\sqrt{\sum_i v_M(f_i)^2} \sqrt{\sum_i v_{C_g}(f_i)^2}} \tag{26}$$

The method of Prager (1999) was evaluated by Lui, Lau, and Baldwin (2014a) in the context of LI over multilingual documents. The cosine similarity was used recently by Schaetti (2017) with TF-IDF values in the language variety identification shared task of PAN 2017 (Rangel et al., 2017b). He compared the performance of the cosine similarity with CNN. For English varieties the cosine similarity achieved 83% accuracy while a CNN reached only 66%.

One common trick with cosine similarity is to pre-normalise the feature vectors to unit length (e.g. Brown, 2012), in which case the calculation takes the form of the simple dot product:

$$R_{\text{dotprod}}(g, M) = \sum_i v_M(f_i) v_{C_g}(f_i) \tag{27}$$

Jauhiainen (2010) used chi-squared distance, calculated as follows:

$$R_{\textit{chi-square}}(g, M) = \sum_i \frac{(v_{C_g}(f_i) - v_M(f_i))^2}{v_M(f_i)} \tag{28}$$

Abainia et al. (2016) compared Manhattan, Bhattacharyya, chi-squared, Canberra, Bray Curtis, histogram intersection, correlation distances, and out-of-place distances, and found the out-of-place method to be the most accurate.

Entropy Singh (2006, 2010) used cross-entropy and symmetric cross-entropy. Cross-entropy is calculated as follows, where $v_M(f_i)$ and $v_{C_g}(f_i)$ are the probabilities of the feature f_i in the the test document M and the corpus C_g:

$$R_{cross\text{-}entropy}(g, M) = \sum_i v_M(f_i) \log v_{C_g}(f_i) \tag{29}$$

Symmetric cross-entropy is calculated as:

$$R_{sym\text{-}cross\text{-}entropy}(g, M) = \sum_i v_M(f_i) \log v_{C_g}(f_i) + v_{C_g}(f_i) \log v_M(f_i) \tag{30}$$

For cross-entropy, distribution M must be smoothed, and for symmetric cross-entropy, both probability distributions must be smoothed. Cross-entropy was used recently by Hanani et al. (2017) with words in the GDI 2017 shared task, but the results were inferior to a byte n-gram based sum of log-frequencies, which attained 7th position in the shared task.

Yamaguchi and Tanaka-Ishii (2012) used a cross-entropy estimating method they call the Mean of Matching Statistics ("MMS"). In MMS, every possible suffix of the test text $u_i, ..., u_{l_M}$ is compared with the language model of each language, and the average of the lengths of the longest possible units in the language model matching the beginning of each suffix is calculated. They compared the performance of MMS with that of PPM (Section 5.7), and find that PPM achieves slightly better performance in their evaluations on over 200 languages.

Sibun and Reynar (1996) and Baldwin and Lui (2010a) calculated the relative entropy between the language models and the test document, as follows:

$$R_{rel\text{-}entropy}(g, M) = \sum_i v_M(f_i) \log \frac{v_M(f_i)}{v_{C_g}(f_i)} \tag{31}$$

This method is also commonly referred to as Kullback-Leibler ("KL") divergence or skew divergence. Jauhiainen (2010) compared relative entropy with the product of the relative frequencies for different-sized character n-grams, and found that relative entropy was only competitive when used with character bigrams. The product of relative frequencies gained clearly higher recall with higher-order n-grams when compared with relative entropy.

Singh (2006, 2010) also used the RE and MRE measures, which are related to relative entropy. The RE measure is calculated as follows:

$$R_{RE}(g, M) = \sum_i v_M(f_i) \frac{\log v_M(f_i)}{\log v_{C_g}(f_i)} \tag{32}$$

MRE is the symmetric version of the same measure. In the tests performed by Singh (2006, 2010), the RE measure with character n-grams outperformed other tested methods obtaining 98.51% precision when classifying 100 character texts between 53 language-encoding pairs.

Logistic Regression (LR) Chen and Maison (2003) used a logistic regression ("LR") model (also commonly referred to as "maximum entropy" within NLP), smoothed with a Gaussian prior (Hosmer, Lemeshow, & Sturdivant, 2013). Porta and Sancho (2014) defined LR for character-based features as follows:

$$R_{LR}(g, M) = \frac{1}{Z} \exp \sum_{j}^{l_{MT}} \sum_{i}^{l_{CF}} v_{C_g}(f_i), \text{ if } \exists f_i \in U(M_j^T) \tag{33}$$

where Z is a normalization factor and l_{MT} is the word count in the word-tokenized test document. Ács et al. (2015) used an LR classifier and found it to be considerably faster than an SVM, with comparable results. Their LR classifier ranked 6 out of 9 on the closed submission track of the DSL 2015 shared task. Lu and Mohamed (2011) used Adaptive Logistic Regression, which automatically optimizes parameters. In recent years LR has been widely used for LI.

Perplexity Ramisch (2008) was the first to use perplexity for LI, in the manner of a language model. He calculated the perplexity for the test document M as follows:

$$H_g(M) = \frac{1}{l_{M^n}} \sum_{i}^{l_{M^n}} \log_2 v_{C_g}(f_i) \tag{34}$$

$$R_{perplexity}(g, M) = 2^{H_g(M)} \tag{35}$$

where $v_{C_g}(f_i)$ were the Katz smoothed relative frequencies of word n-grams f_i of the length n. Rodrigues (2012) and Jauhiainen et al. (2017b) evaluated the best performing method used by Vatanen et al. (2010). Character n-gram based perplexity was the best method for extremely short texts in the evaluations of Jauhiainen et al. (2017b), but for longer sequences the methods of Brown (2012) and Jauhiainen (2010) proved to be better. Lately, Gamallo et al. (2017) used perplexity in the DSL and GDI 2017 shared tasks, achieving 9th and 8th place, respectively (Zampieri et al., 2017).

Other similarity measures Rau (1974) used Yule's characteristic K and the Kolmogorov-Smirnov goodness of fit test to categorize languages. Kolmogorov-Smirnov proved to be the better of the two, obtaining 89% recall for 53 characters (one punch card) of text when choosing between two languages. In the goodness of fit test, the ranks of features in the models of the languages and the test document are compared. Martins and Silva (2005) experimented with Jiang and Conrath's (JC) distance (Jiang & Conrath, 1997) and Lin's similarity measure (Lin, 1998), as well as the out-of-place method. They conclude that Lin's similarity measure was consistently the most accurate of the three. JC-distance measure was later evaluated by Singh (2006, 2010), and was outperformed by the RE measure. Ranaivo-Malançon and Ng (2005) and Ranaivo-Malançon (2006) calculated special ratios from the number of trigrams in the language models when compared with the text to be identified. da Silva and Lopes (2006a, 2006b, 2007) used the quadratic discrimination score to create the feature vectors representing the languages and the test document. They then calculated the Mahalanobis distance between the languages and the test document. Their language identifier obtained 98.9% precision when classifying texts of four "screen lines"

between 19 languages. Nguyen and Cornips (2016) used odds ratio to identify the language of parts of words when identifying between two languages. Odds ratio for language g when compared with language h for morph f_i is calculated as in Equation 36.

$$R_{odds}(g, f_i) = \log \frac{v_{C_h}(f_i)(1 - v_{C_g}(f_i))}{(1 - v_{C_h}(f_i))v_{C_g}(f_i)} \tag{36}$$

6.7 Discriminant Functions

The differences between languages can be stored in discriminant functions. The functions are then used to map the test document into an n-dimensional space. The distance of the test document to the languages known by the language identifier is calculated, and the nearest language is selected (in the manner of a nearest prototype classifier).

Murthy and Kumar (2006) used multiple linear regression to calculate discriminant functions for two-way LI for Indian languages. Bhargava, Sharma, Sharma, and Baid (2015) compared linear regression, NB, and LR. The precision for the three methods was very similar, with linear regression coming second in terms of precision after LR.

Multiple discriminant analysis was used for LI by Mustonen (1965). He used two functions, the first separated Finnish from English and Swedish, and the second separated English and Swedish from each other. He used Mahalanobis' D^2 as a distance measure. Vinosh Babu and Baskaran (2005) used Multivariate Analysis ("MVA") with Principal Component Analysis ("PCA") for dimensionality reduction and LI. Takçı and Ekinci (2012) compared discriminant analysis with SVM and NN using characters as features, and concluded that the SVM was the best method.

King and Abney (2013) experimented with the Winnow 2 algorithm (Littlestone, 1987), but the method was outperformed by other methods they tested.

6.8 Support Vector Machines ("SVMs")

With support vector machines ("SVMs"), a binary classifier is learned by learning a separating hyperplane between the two classes of instances which maximizes the margin between them (Boser, Guyon, & Vapnik, 1992). The simplest way to extend the basic SVM model into a multiclass classifier is via a suite of one-vs-rest classifiers, where the classifier with the highest score determines the language of the test document. One feature of SVMs that has made them particularly popular is their compatibility with kernels, whereby the separating hyperplane can be calculated via a non-linear projection of the original instance space. In the following paragraphs, we list the different kernels that have been used with SVMs for LI.

Linear kernel SVMs For LI with SVMs, the predominant approach has been a simple linear kernel SVM model. The linear kernel model has a weight vector $v_{C_g}(f)$ and the classification of a feature vector $v_M(f)$, representing the test document M, is calculated as follows:

$$R_{\text{svm-lin}}(g, M) = (\sum_i v_M(f_i)v_{C_g}(f_i)) + b \tag{37}$$

where b is a scalar bias term. If $R_{\text{svm-lin}}$ is equal to or greater than zero, M is categorized as g.

The first to use a linear kernel SVM were Kim and Park (2007), and generally speaking, linear-kernel SVMs have been widely used for LI, with great success across a range of shared tasks. Goutte et al. (2014) won the DSL 2014 shared task. The three submissions by Malmasi and Dras (2015b) to the DSL 2015 shared task, using three different linear SVM configurations, were the three best submissions overall. The second (Zampieri et al., 2015) and the third (Goutte & Léger, 2015) best teams also utilized SVMs with a linear kernel. Çöltekin and Rama (2016) won the DSL 2016 and Bestgen (2017) the DSL 2017 shared task using linear SVMs. Malmasi and Zampieri (2016) won ADI 2016 and Malmasi and Zampieri (2017b) won the GDI 2017 shared task, with an ensemble of linear kernel SVMs.

Polynomial kernel SVMs Bar and Dershowitz (2014) were the first to apply polynomial kernel SVMs to LI. With a polynomial kernel $R_{\text{svm-pol}}$ can be calculated as:

$$R_{\text{svm-pol}}(g, M) = ((\sum_i v_M(f_i) v_{C_g}(f_i)) + b)^d \tag{38}$$

where d is the polynomial degree, and a hyperparameter of the model.

Radial Basis Function (RBF) kernel SVMs Another popular kernel is the RBF function, also known as a Gaussian or squared exponential kernel. With an RBF kernel $R_{\text{svm-rbf}}$ is calculated as:

$$R_{\text{svm-rbf}}(g, M) = \exp(-\frac{(\sum_i |v_M(f_i) - v_{C_g}(f_i)|)^2}{2\sigma^2}) \tag{39}$$

where σ is a hyperparameter. Botha et al. (2007) were the first to use an RBF kernel SVM for LI. In their experiments, they found that the SVM was more accurate than a probability difference based classifier, but with substantially higher computational complexity.

Sigmoid kernel SVMs With sigmoid kernel SVMs, also known as hyperbolic tangent SVMs, $R_{\text{svm-sig}}$ can be calculated as:

$$R_{\text{svm-sig}}(g, M) = \tanh((\sum_i v_M(f_i) v_{C_g}(f_i)) + b) \tag{40}$$

Bhargava and Kondrak (2010) were the first to use a sigmoid kernel SVM for LI, followed by Majliš (2012b), who found the SVM to perform better than NB, Classification And Regression Tree ("CART"), or the sum of relative frequencies.

Other kernels Other kernels that have been used with SVMs for LI include exponential kernels (Alrifai et al., 2017) and rational kernels (Porta, 2014). Kruengkrai et al. (2005) were the first to use SVMs for LI, in the form of string kernels using Ukkonen's algorithm. They used the same string kernels with Euclidean distance, which did not perform as well as SVM. Castro et al. (2017) compared SVMs with linear and on-line passive–aggressive kernels for LI, and found passive–aggressive kernels to perform better, but both SVMs to be inferior to NB and Log-Likelihood Ratio (sum of log-probabilities). Kim and Park (2007) experimented with the Sequential Minimal Optimization ("SMO") algorithm, but found a simple linear kernel SVM to perform better. Alshutayri et al. (2016) achieved the

Reference	linear	string	RBF	sigmoid	d^n	exp.	pas. aggr.
Bhargava and Kondrak (2010)	***	*	**	**	*		
Takçı and Güngör (2012)	*		*	*			
Giwa and Davel (2013), Giwa (2016)	**		***				
Eldesouki et al. (2016)	***	*					
Hanani et al. (2016)	***		*		*		
Xu et al. (2016)	***		*	*	*		
Alrifai et al. (2017)	**				***	*	
Castro et al. (2017)	**						***
Franco-Salvador et al. (2017a)	*		*		*		

Table 10: References where SVMs have been tested with different kernels. The columns indicate the kernels used. "d^n" stands for polynomial kernel.

best results using the SMO algorithm, whereas Lamabam and Chakma (2016) found CRFs to work better than SMO. Alrifai et al. (2017) found that SMO was better than linear, exponential and polynomial kernel SVMs for Arabic tweet gender and dialect prediction.

Table 10 lists articles where SVMs with different kernels have been compared. Goutte, Léger, Malmasi, and Zampieri (2016) evaluated three different SVM approaches using datasets from different DSL shared tasks. SVM-based approaches were the top performing systems in the 2014 and 2015 shared tasks.

Margin Infused Relaxed Algorithm ("MIRA") Williams and Dagli (2017) used SVMs with the Margin Infused Relaxed Algorithm, which is an incremental version of SVM training. In their evaluation, this method achieved better results than off-the-shelf `langid.py` (Lui & Baldwin, 2011).

6.9 Neural Networks ("NN")

Batchelder (1992) was the first to use Neural Networks ("NN") for LI, in the form of a simple BackPropagation Neural Network ("BPNN") (Hecht-Nielsen, 1989) with a single layer of hidden units, which is also called a multi-layer perceptron ("MLP") model. She used words as the input features for the neural network. Tian, Häkkinen, Riis, and Jensen (2002) and Tian and Suontausta (2003) succesfully applied MLP to LI.

Jalam and Teytaud (2001b, 2001a) and Jalam (2003) used radial basis function (RBF) networks for LI. Selamat, Ng, and Mikami (2007) were the first to use adaptive resonance learning ("ART") neural networks for LI. Abainia et al. (2016) used Neural Text Categorizer ("NTC": Jo, 2008) as a baseline. NTC is an MLP-like NN using string vectors instead of number vectors.

MacNamara et al. (1998) were the first to use a RNN (Jordan, 1986) for LI. They concluded that RNNs are less accurate than the simple sum of logarithms of counts of character bi- or trigrams, possibly due to the relatively modestly-sized dataset they experimented with. Babu and Kumar (2010) compared NNs with the out-of-place method (see sec. 6.6). Their results show that the latter, used with bigrams and trigrams of characters, obtains clearly higher identification accuracy when dealing with test documents shorter than 400 characters.

RNNs were more successfully used later by Chang and Lin (2014) who also incorporated character n-gram features in to the network architecture. Cazamias et al. (2015) were the first to use a Long Short-Term Memory ("LSTM") for LI (Hochreiter & Schmidhuber, 1997), and Bjerva (2016) was the first to use Gated Recurrent Unit networks ("GRUs"), both of which are RNN variants. Bjerva (2016) used byte-level representations of sentences as input for the networks. Recently, Hanani et al. (2016) and Samih et al. (2016) also used LSTMs. Samih et al. (2016) participated in the shared tasks of the 2nd workshop on Computational Approaches to Code Switching, and attained first place for MSA-Egyptian and second place for Spanish–English. Later, GRUs were successfully used for LI by Jurgens, Tsvetkov, and Jurafsky (2017) and Kocmi and Bojar (2017). In addition to GRUs, Bjerva (2016) also experimented with deep residual networks ("ResNets") at DSL 2016.

During 2016 and 2017, there was a spike in the use of convolutional neural networks (CNNs) for LI, most successfully by Jaech et al. (2016b) and Jaech, Mulcaire, Hathi, Ostendorf, and Smith (2016a). Recently, Li, Cohn, and Baldwin (2018) combined a CNN with adversarial learning to better generalize to unseen domains, surpassing the results of Lui and Baldwin (2012) based on the same training regime as `langid.py`.

Medvedeva et al. (2017) used CBOW NN, achieving better results over the development set of DSL 2017 than RNN-based neural networks. Franco-Salvador et al. (2017b) used deep averaging networks (DANs) based on word embeddings in language variety identification. They participated in the language variety identification track of the PAN 2017 Author Profiling Shared Task (Rangel et al., 2017b) and attained 14th place with an average accuracy of 85.1%.

6.10 Other Methods

Simaki et al. (2017) used the decision table (Kohavi, 1995) majority classifier algorithm from the WEKA toolkit in English variety detection. In their tests, the decision table achieved 73.07% accuracy, not far behind Bagging (see Section 6.11), which was the best method they tested.

Ueda and Nakagawa (1990) were the first to apply hidden Markov models (HMM) to LI (Rabiner & Juang, 1986). More recently HMMs have been used by Adouane and Dobnik (2017), Guzmán et al. (2017), and Rijhwani et al. (2017). Binas (2005) generated aggregate Markov models, which resulted in the best results when distinguishing between six languages, obtaining 74% accuracy with text length of ten characters. King et al. (2014) used an extended Markov Model ("eMM"), which is essentially a standard HMM with modified emission probabilities. Their eMM used manually optimized weights to combine four n-gram scores (products of relative frequencies) into one n-gram score. Xia, Lewis, and Poon (2009) used Markov logic networks (Richardson & Domingos, 2006) to predict the language used in interlinear glossed text examples contained in linguistic papers.

Hayta, Takçı, and Eminli (2013) evaluated the use of unsupervised Fuzzy C Means algorithm ("FCM") in language identification (Bezdek, Ehrlich, & Full, 1984). The unsupervised algorithm was used on the training data to create document clusters. Each cluster was tagged with the language having the most documents in the cluster. Then in the identification phase, the test text was mapped to the closest cluster and identified with its

language. A supervised centroid classifier based on cosine similarity obtained clearly better results in their experiments (93% vs. 77% accuracy).

Barbaresi (2016) and Martinc et al. (2017) evaluated the extreme gradient boosting ("XGBoost") method (Chen & Guestrin, 2016). Barbaresi (2016) found that gradient boosting gave better results than RFs, while conversely, Martinc et al. (2017) found that LR gave better results than gradient boosting.

Benedetto, Caglioti, and Loreto (2002) used compression methods for LI, whereby a single test document is added to the training text of each language in turn, and the language with the smallest difference (after compression) between the sizes of the original training text file and the combined training and test document files is selected as the prediction. This has obvious disadvantages in terms of real-time computational cost for prediction, but is closely related to language modeling approaches to LI (with the obvious difference that the language model does not need to be retrained for each test document). In terms of compression methods, Haţegan, Bârligă, and Tăbuş (2009) experimented with Maximal Tree Machines ("MTMs"), and Bush (2014) used LZW-based compression.

Very popular in text categorization and topic modeling, Tratz, Briesch, Laoudi, and Voss (2013), Tratz (2014), and Voss et al. (2014) used Latent Dirichlet Allocation ("LDA": Blei et al., 2003) based features in classifying tweets between Arabic dialects, English, and French. Each tweet was assigned with an LDA topic, which was used as one of the features of an LR classifier.

Poulston et al. (2017) used a Gaussian Process classifier with an RBF kernel in an ensemble with an LR classifier. Their ensemble achieved only ninth place in the PAN Author Profiling language variety shared task (Rangel et al., 2017b) and did not reach the results of the baseline for the task.

Espichán-Linares and Oncevay-Marcos (2017, 2018) used a Passive Aggressive classifier (Crammer, Dekel, Keshet, Shalev-Shwartz, & Singer, 2006), which proved to be almost as good as the SVMs in their evaluations between five different machine learning algorithms from the same package.

6.11 Ensemble Methods

Ensemble methods are meta-classification methods capable of combining several base classifiers into a combined model via a "meta-classifier" over the outputs of the base classifiers, either explicitly trained or using some heuristic. It is a simple and effective approach that is used widely in machine learning to boost results beyond those of the individual base classifiers, and particularly effective when applied to large numbers of individually uncorrelated base classifiers.

Majority and Plurality Voting Rau (1974) used simple majority voting to combine classifiers using different features and methods. In majority voting, the language of the test document is identified if a majority ($> \frac{1}{2}$) of the classifiers in the ensemble vote for the same language. In plurality voting, the language with most votes is chosen as in the simple scoring method (Equation 17). Some authors also refer to plurality voting as majority voting.

Carter, Weerkamp, and Tsagkias (2013) used majority voting in tweet LI. Giwa and Davel (2014) used majority voting with JSM classifiers. Goutte et al. (2014) and Malmasi

and Dras (2015a) used majority voting between SVM classifiers trained with different features. Gupta et al. (2014) used majority voting to combine four classifiers: RF, random tree, SVM, and DT. Doval, Vilares, and Vilares (2014) and Lui and Baldwin (2014) used majority voting between three off-the-shelf language identifiers. Leidig (2014) used majority voting between perplexity-based and other classifiers. Zamora et al. (2014) used majority voting between three sum of relative frequencies-based classifiers where values were weighted with different weighting schemes. Malmasi and Dras (2015b) evaluated six ensemble methods with SVMs, and found that the mean probability rule performed better than plurality voting. Later, Malmasi and Dras (2017) confirmed these earlier findings. Also, Malmasi and Zampieri (2016, 2017a, 2017b) and Mendoza and Mendelsohn (2017) used plurality voting with SVMs.

Gamallo et al. (2017) used voting between several perplexity-based classifiers with different features at the 2017 DSL shared task. A voting ensemble gave better results on the closed track than a singular word-based perplexity classifier (0.9025 weighted F1-score over 0.9013), but worse results on the open track (0.9016 with ensemble and 0.9065 without).

Highest Probability Ensemble In a highest probability ensemble, the winner is simply the language which is given the highest probability by any of the individual classifiers in the ensemble. You et al. (2008) used Gaussian Mixture Models ("GMM") to give probabilities to the outputs of classifiers using different features. Doval et al. (2014) used higher confidence between two off-the-shelf language identifiers. Goutte et al. (2014) used GMM to transform SVM prediction scores into probabilities. Malmasi and Dras (2015b, 2017) used highest confidence over a range of base SVMs. Malmasi and Dras (2017) used an ensemble composed of low-dimension hash-based classifiers. According to their experiments, hashing provided up to 86% dimensionality reduction without negatively affecting performance. Their probability-based ensemble obtained 89.2% accuracy, while the voting ensemble got 88.7%. Balažević et al. (2016) combined an SVM and a LR classifier.

Mean Probability Rule A mean probability ensemble can be used to combine classifiers that produce probabilities (or other mutually comparable values) for languages. The average of values for each language over the classifier results is used to determine the winner and the results are equal to the sum of values method (Equation 18). Malmasi and Dras (2015b) evaluated several ensemble methods and found that the mean probability ensemble attained better results than plurality voting, median probability, product, highest confidence, or Borda count ensembles. With the mean probability ensemble which combined several SVM classifiers, Malmasi and Dras (2015b) outperformed all the other participating teams on the DSL 2015 shared task.

Median Probability Rule In a median probability ensemble, the medians over the probabilities given by the individual classifiers are calculated for each language. Malmasi and Dras (2015b) and Malmasi and Zampieri (2016) used a median probability rule ensemble over SVM classifiers. Consistent with the results of Malmasi and Dras (2015b), Malmasi and Zampieri (2016) found that a mean ensemble was better than a median ensemble, attaining 68% accuracy vs. 67% for the median ensemble.

Product Rule A product rule ensemble takes the probabilities for the base classifiers and calculates their product (or sum of the log probabilities), with the effect of penalising any

language where there is a particularly low probability from any of the base classifiers. Giwa and Davel (2014) used log probability voting with JSM classifiers. Giwa and Davel (2014) observed a small increase in average accuracy using the product ensemble over a majority voting ensemble.

k-best Ensemble (k-best) In a k-best ensemble, several models are created for each language g by partitioning the corpus C_g into separate samples. The score $R(C_{g_i}, M)$ is calculated for each model. For each language, plurality voting is then applied to the k models with the best scores to predict the language of the test document M. Jalam and Teytaud (2001a) evaluated k-best with $k = 1$ based on several similarity measures. Kerwin (2006) compared $k = 10$ and $k = 50$ and concluded that there was no major difference in accuracy when distinguishing between six languages (100 character test set). Baykan, Henzinger, and Weber (2008) experimented with k-best classifiers, but they gave clearly worse results than the other classifiers they evaluated. Barman et al. (2014b) used k-best in two phases, first selecting $k_1 = 800$ closest neighbors with simple similarity, and then using $k_2 = 16$ with a more advanced similarity ranking.

Bootstrap Aggregating (Bagging) In bagging, independent samples of the training data are generated by random sampling with replacement, individual classifiers are trained over each such training data sample, and the final classification is determined by plurality voting (Breiman, 1996). Martinc et al. (2017) evaluated the use of bagging with an LR classifier in PAN 2017 language variety identification shared task (Rangel et al., 2017b), however, bagging did not improve the accuracy in the 10-fold cross-validation experiments on the training set. Sierra, Montes-y Gómez, Solorio, and González (2017) used bagging with word convolutional neural networks ("W-CNN") in the same shared task, attaining 7th position. Simaki et al. (2017) used bagging with DTs in English national variety detection and found DT-based bagging to be the best evaluated method when all 60 different features (a wide selection of formal, POS, lexicon-based, and data-based features) were used, attaining 73.86% accuracy. Simaki et al. (2017) continued the experiments using the ReliefF feature selection algorithm from the WEKA toolkit to select the most efficient features, and achieved 77.32% accuracy over the reduced feature set using a NB classifier.

Rotation Forest Rangel et al. (2017a) evaluated the Rotation Forest (Rodríguez, Kuncheva, & Alonso, 2006) meta classifier for DTs. The method randomly splits the used features into a pre-determined number of subsets and then uses PCA for each subset. It obtained 66.6% accuracy, attaining fifth place among the twelve methods evaluated.

Adaptive Boosting (AdaBoost) The AdaBoost algorithm (Freund & Schapire, 1997) examines the performance of the base classifiers on the evaluation set and iteratively boosts the significance of misclassified training instances, with a restart mechanism to avoid local minima. AdaBoost was the best of the five machine learning techniques evaluated by van der Lee and Bosch (2017), faring better than C4.5 (Quinlan, 1993), NB, RF, and linear SVM. Rangel et al. (2017a) used the LogitBoost variation of AdaBoost (Friedman, Hastie, & Tibshirani, 2000). It obtained 67.0% accuracy, attaining third place among the twelve methods evaluated.

Stacked generalization (Stacking) In stacking, a higher level classifier is explicitly trained on the output of several base classifiers (Wolpert, 1992). You et al. (2008) used

AdaBoost.ECC and CART to combine classifiers using different features. More recently, Mathur et al. (2017) used LR to combine the results of five RNNs. As an ensemble they produced better results than NB and LR, which were better than the individual RNNs. Also in 2017, Malmasi and Zampieri (2017a, 2017b) used RF to combine several linear SVMs with different features. The system used by Malmasi and Zampieri (2017b) ranked first in the German dialect identification shared task, and the system by Malmasi and Zampieri (2017a) came second (71.65% accuracy) in the Arabic dialect identification shared task.

7. Empirical Evaluation

In the previous two sections, we have alluded to issues of evaluation in LI research to date. In this section, we examine the literature more closely, providing a broad overview of the evaluation metrics that have been used, as well as the experimental settings in which LI research has been evaluated.

7.1 Standardized Evaluation for LI

The most common approach is to treat the task as a document-level classification problem. Given a set of evaluation documents, each having a known correct label from a closed set of labels (often referred to as the "gold-standard"), and a predicted label for each document from the same set, the document-level accuracy is the proportion of documents that are correctly labeled over the entire evaluation collection. This is the most frequently reported metric and conveys the same information as the error rate, which is simply the proportion of documents that are incorrectly labeled (i.e. $1 - \text{accuracy}$).

Authors sometimes provide a per-language breakdown of results. There are two distinct ways in which results are generally summarized per-language: (1) precision, in which documents are grouped according to their predicted language; and (2) recall, in which documents are grouped according to what language they are actually written in. Earlier work has tended to only provide a breakdown based on the correct label (i.e. only reporting per-language recall). This gives us a sense of how likely a document in any given language is to be classified correctly, but does not give an indication of how likely a prediction for a given language is of being correct. Under the monolingual assumption (i.e. each document is written in exactly one language), this is not too much of a problem, as a false negative for one language must also be a false positive for another language, so precision and recall are closely linked. Nonetheless, authors have recently tended to explicitly provide both precision and recall for clarity. It is also common practice to report an F-score F, which is the harmonic mean of precision and recall. The F-score (also sometimes called F1-score or F-measure) was developed in IR to measure the effectiveness of retrieval with respect to a user who attaches different relative importance to precision and recall (van Rijsbergen, 1979). When used as an evaluation metric for classification tasks, it is common to place equal weight on precision and recall (hence "F1"-score, in reference to the β hyper-parameter, which equally weights precision and recall when $\beta = 1$).

In addition to evaluating performance for each individual language, authors have also sought to convey the relationship between classification errors and specific sets of languages. Errors in LI systems are generally not random; rather, certain sets of languages are much more likely to be confused. The typical method of conveying this information is through

the use of a confusion matrix, a tabulation of the distribution of (predicted language, actual language) pairs.

Presenting full confusion matrices becomes problematic as the number of languages considered increases, and as a result has become relatively uncommon in work that covers a broader range of languages. Per-language results are also harder to interpret as the number of languages increases, and so it is common to present only collection-level summary statistics. There are two conventional methods for summarizing across a whole collection: (1) giving each document equal weight; and (2) giving each class (i.e. language) equal weight. (1) is referred to as a micro-average, and (2) as a macro-average. For LI under the monolingual assumption, micro-averaged precision and recall are the same, since each instance of a false positive for one language must also be a false negative for another language. In other words, micro-averaged precision and recall are both simply the collection-level accuracy. On the other hand, macro-averaged precision and recall give equal weight to each language. In datasets where the number of documents per language is the same, this again works out to being the collection-level average. However, LI research has frequently dealt with datasets where there is a substantial skew between classes. In such cases, the collection-level accuracy is strongly biased towards more heavily-represented languages. To address this issue, in work on skewed document collections, authors tend to report both the collection-level accuracy and the macro-averaged precision/recall/F-score, in order to give a more complete picture of the characteristics of the method being studied.

Whereas the notions of macro-averaged precision and recall are clearly defined, there are two possible methods to calculate the macro-averaged F-score. The first is to calculate it as the harmonic mean of the macro-averaged precision and recall, and the second is to calculate it as the arithmetic mean of the per-class F-score.

The comparability of published results is also limited by the variation in size and source of the data used for evaluation. In work to date, authors have used data from a variety of different sources to evaluate the performance of proposed solutions. Typically, data for a number of languages is collected from a single source (for example web pages: Lins & Gonçalves, 2004, or a newswire source: Takçı & Güngör, 2012), and the number of languages considered varies widely. Earlier work tended to focus on a smaller number of Western European languages. Notable exceptions are Sibun and Spitz (1994), which considered 23 languages including some African and Asian languages, and Kikui (1996), which included Japanese. Later work has shifted focus to supporting larger numbers of languages simultaneously, with the work of Brown (2014b) pushing the upper bound, reporting a language identifier that supports over 1300 languages. The increased size of the language set considered is partly due to the increased availability of language-labeled documents from novel sources such as Wikipedia and Twitter. This supplements existing data from translations of the Universal Declaration of Human Rights, bible translations, as well as parallel texts from MT datasets such as OPUS (Tiedemann, 2012) and SETimes, and European Government data such as JRC-Acquis (Steinberger et al., 2006). These factors have led to a shift away from proprietary datasets such as the ECI multilingual corpus that were commonly used in earlier research. As more languages are considered simultaneously, the accuracy of LI systems decreases. A particularly striking illustration of this is the evaluation results by Jauhiainen et al. (2017b) for the logLIGA method (Vogel & Tresner-Kirsch, 2012). Vogel and Tresner-Kirsch (2012) report an accuracy of 99.8% over tweets (averaging

80 characters) in six European languages as opposed to the 97.9% from the original LIGA method. The LIGA and logLIGA implementations by Jauhiainen et al. (2017b) have comparable accuracy for six languages, but the accuracy for 285 languages (with 70 character test length) is only slightly over 60% for logLIGA and the original LIGA method is at almost 85%. Many evaluations are not directly comparable as the test sizes, language sets, and hyper-parameters differ. A particularly good example is the method of Cavnar and Trenkle (1994). The original paper reports an accuracy of 99.8% over eight European languages (>300 bytes test size). Lui and Baldwin (2011) report an accuracy of 68.6% for the method over a dataset of 67 languages (500 byte test size), and Jauhiainen et al. (2017b) report an accuracy of over 90% for 285 languages (25 character test size).

Separate to the question of the number and variety of languages included are issues regarding the quantity of training data used. A number of studies have examined the relationship between LI accuracy and quantity of training data through the use of learning curves. The general finding is that LI accuracy increases with more training data, though there are some authors that report an optimal amount of training data, where adding more training data decreases accuracy thereafter (Ljubešić, Mikelić, & Boras, 2007). Overall, it is not clear whether there is a universal quantity of data that is "enough" for any language, rather this amount appears to be affected by the particular set of languages as well as the domain of the data. As a rough gauge, training data quantities on the order of 50KB per language onwards appear to be quite common (Souter et al., 1994; Adams & Resnik, 1997; Prager, 1999; Xafopoulos et al., 2004; Takçı & Güngör, 2012), and most of the reported learning curves appear to plateau somewhere around 100KB to 1MB per language. The breakdown presented by Baldwin and Lui (2010a) shows that with less than 100KB per language, there are some languages where classification accuracy is near perfect, whereas there are others where it is very poor.

Another aspect that is frequently reported on is how long a sample of text needs to be before its language can be correctly detected. Unsurprisingly, the general consensus is that longer samples are easier to classify correctly. There is a strong interest in classifying short segments of text, as certain applications naturally involve short text documents, such as LI of microblog messages or search engine queries. Another area where LI of texts as short as one word has been investigated is in the context of dealing with documents that contain text in more than one language, where word-level LI has been proposed as a possible solution (see Section 10.6). These outstanding challenges have led to research focused specifically on LI of shorter segments of text, which we discuss in more detail in Section 10.7.

From a practical perspective, knowing the rate at which a LI system can process and classify documents is useful as it allows a practitioner to predict the time required to process a document collection given certain computational resources. However, so many factors influence the rate at which documents are processed that comparison of absolute values across publications is largely meaningless. Instead, it is more valuable to consider publications that compare multiple systems under controlled conditions (same computer hardware, same evaluation data, etc.). The most common observations are that classification times between different algorithms can differ by orders of magnitude (Poutsma, 2002; Lui & Baldwin, 2011, 2012; Majliš, 2012b; Takçı & Ekinci, 2012; Takçı & Güngör, 2012; Brown, 2013), and that the fastest methods are not always the most accurate (Poutsma, 2002; Majliš,

2012b; Brown, 2013). Beyond that, the diversity of systems tested and the variety in the test data make it difficult to draw further conclusions about the relative speed of algorithms.

Where explicit feature selection is used, the number of features retained is a parameter of interest, as it affects both the memory requirements of the LI system and its classification rate. In general, a smaller feature set results in a faster and more lightweight identifier. Relatively few authors give specific details of the relationship between the number of features selected and accuracy (Cavnar & Trenkle, 1994; Jauhiainen, 2010; Vatanen et al., 2010; Brown, 2012; Majliš, 2012b; Jhamtani et al., 2014). A potential reason for this is that the improvement in accuracy plateaus with increasing feature count (Vatanen et al., 2010; Lui & Baldwin, 2011; Brown, 2012; Lui & Baldwin, 2012), though the exact number of features required varies substantially with the method and the data used. At the lower end of the scale, Cavnar and Trenkle (1994) report that 300–400 features per language is sufficient. Conversely Jauhiainen et al. (2017b) found that, for the same method, the best results for the evaluation set were attained with 20,000 features per language.

7.2 Corpora Used for LI Evaluation

As discussed in Section 7.1, the objective comparison of different methods for LI is difficult due to the variation in the data that different authors have used to evaluate LI methods. Baldwin and Lui (2010a) emphasize this by demonstrating how the performance of a system can vary according to the data used for evaluation. This implies that comparisons of results reported by different authors may not be meaningful, as a strong result in one paper may not translate into a strong result on the dataset used in a different paper. In other areas of research, authors have proposed standardized corpora to allow for the objective comparison of different methods.

Some authors have released datasets to accompany their work, to allow for direct replication of their experiments and encourage comparison and standardization. Table 11 lists a number of datasets that have been released to accompany specific LI publications. In this list, we only include corpora that were prepared specifically for LI research, and that include the full text of documents. Corpora of language-labelled Twitter messages that only provide document identifiers are also available, but reproducing the full original corpus is always an issue as the original Twitter messages are deleted or otherwise made unavailable.

One challenge in standardizing datasets for LI is that the codes used to label languages are not fully standardized, and a large proportion of labeling systems only cover a minor portion of the languages used in the world today (Constable & Simons, 2000). Xia, Lewis, and Lewis (2010) discuss this problem in detail, listing different language code sets, as well as the internal structure exhibited by some of the code sets. Some standards consider certain groups of "languages" as varieties of a single macro-language, whereas others consider them to be discrete languages. An example of this is found in South Slavic languages, where some language code sets refer to Serbo-Croatian, whereas others make distinctions between Bosnian, Serbian and Croatian (Tiedemann & Ljubešić, 2012). The unclear boundaries between such languages make it difficult to build a reference corpus of documents for each language, or to compare language-specific results across datasets.

Another challenge in standardizing datasets for LI is the great deal of variation that can exist between data in the same language. We examine this in greater detail in Section 10.2,

Reference	Type	Source
Baldwin and Lui (2010a)	Multilingual (81)	Government Documents, News Texts, Wikipedia
`https://github.com/varh1i/language_detection/tree/master/src/main/resources/naacl2010-langid`		
Baldwin and Lui (2010b)	Multilingual (74)	Wikipedia (synthetic multilingual docs)
`http://people.eng.unimelb.edu.au/tbaldwin/etc/altw2010-langid.tgz`		
Vatanen et al. (2010)	Multiglingual (281)	Universal Declaration of Human Rights (UDHR)
`http://research.ics.aalto.fi/cog/data/udhr/`		
Tromp and Pechenizkiy (2011)	Multilingual (6)	Twitter
`http://www.win.tue.nl/~mpechen/projects/smm/LIGA_Benelearn11_dataset.zip`		
Zaidan and Callison-Burch (2011)	Arabic dialects (5)	Arabic Online Commentary (AOC)
`https://github.com/sjeblee/AOC`		
Lui and Baldwin (2011)	Multilingual	Various
`http://people.eng.unimelb.edu.au/tbaldwin/etc/ijcnlp2011-langid.tgz`		
Majliš (2012b)	Multilingual (124)	Wikipedia (YALI)
`http://ufal.mff.cuni.cz/tools/yali`		
Tiedemann and Ljubešić (2012)	Multilingual (10)	SETimes (News Texts)
`http://www.nljubesic.net/resources/corpora/setimes/`		
Brown (2013)	Multilingual (970/1279)	Bible Translations, Wikipedia
`http://sourceforge.net/projects/la-strings/files/Language-Data/`		
King and Abney (2013)	Multilingual (30)	Web Crawl
`http://www-personal.umich.edu/~benking/resources/mixed-language-annotations-release-v1.0.tgz`		
Lui and Baldwin (2014)	Multilingual (65)	Twitter
`http://people.eng.unimelb.edu.au/tbaldwin/data/lasm2014-twituser-v1.tgz`		
Lui et al. (2014a)	Multilingual (156)	WikipediaMulti
`http://people.eng.unimelb.edu.au/tbaldwin/etc/wikipedia-multi-v6.tgz`		
Tan et al. (2014)	Multilingual (22)	News Texts
`http://ttg.uni-saarland.de/resources/DSLCC/`		
Chanda et al. (2016b)	Code-switching (2)	Facebook chats
`https://github.com/ArunavhaChanda/Facebook-Code-Mixed-Corpus`		
Blodgett, Wei, and O'Connor (2017)	Multilingual (70)	Twitter70
`http://slanglab.cs.umass.edu/TwitterLangID/`		

Table 11: Published LI Datasets

where we discuss how the same language can use a number of different orthographies, can be digitized using a number of different encodings, and may also exist in transliterated forms. The issue of variation within a language complicates the development of standardized datasets, due to challenges in determining which variants of a language should be included. Since we have seen that the performance of LI systems can vary per-domain (Baldwin & Lui, 2010a), that LI research is often motivated by target applications (see Section 8), and that domain-specific information can be used to improve accuracy (see Section 10.9), it is often unsound to use a generic LI dataset to develop a language identifier for a particular domain.

A third challenge in standardizing datasets for LI is the cost of obtaining correctly-labeled data. Manual labeling of data is usually prohibitively expensive, as it requires access to native speakers of all languages that the dataset aims to include. Large quantities of raw text data are available from sources such as web crawls or Wikipedia, but this data is frequently mislabeled (e.g. most non-English Wikipedias still include some English-language

documents). In constructing corpora from such resources, it is common to use some form of automatic LI, but this makes such corpora unsuitable for evaluation purposes as they are biased towards documents that can be correctly identified by automatic systems (Lui & Baldwin, 2014). Future work in this area could investigate other means of ensuring correct gold-standard labels while minimizing the annotation cost.

Despite these challenges, standardized datasets are critical for replicable and comparable research in LI. Where a subset of data is used from a larger collection, researchers should include details of the specific subset, including any breakdown into training and test data, or partitions for cross-validation. Where data from a new source is used, justification should be given for its inclusion, as well as some means for other researchers to replicate experiments on the same dataset.

7.3 LI Shared Tasks

To address specific sub-problems in LI, a number of shared tasks have been organized on problems such as LI in multilingual documents (Baldwin & Lui, 2010b), code-switched data (Solorio et al., 2014), discriminating between closely related languages (Zampieri, Tan, Ljubešić, & Tiedemann, 2014), and dialect and language variety identification in various languages (Grouin, Forest, Da Sylva, Paroubek, & Zweigenbaum, 2011; Zampieri et al., 2017; Rangel et al., 2017b; Ali, Vogel, & Renals, 2017). Shared tasks are important for LI because they provide datasets and standardized evaluation methods that serve as benchmarks for the LI community. We summarize all LI shared tasks organized to date in Table 12.

Generally, datasets for shared tasks have been made publicly available after the conclusion of the task, and are a good source of standardized evaluation data. However, the shared tasks to date have tended to target specific sub-problems in LI, and no general, broad-coverage LI datasets have been compiled. Widespread interest in LI over closely-related languages has resulted in a number of shared tasks that specifically tackle the issue. Some tasks have focused on varieties of a specific language. For example, the DEFT2010 shared task (Grouin et al., 2011) examined varieties of French, requiring participants to classify French documents with respect to their geographical source, in addition to the decade in which they were published. Another example is the Arabic Dialect Identification ("ADI") shared task at the VarDial workshop (Malmasi, 2017; Zampieri et al., 2017), and the Arabic Multi-Genre Broadcast ("MGB") Challenge (Ali et al., 2017).

Two shared tasks focused on a narrow group of languages using Twitter data. The first was TweetLID, a shared task on LI of Twitter messages according to six languages in common use in Spain, namely: Spanish, Portuguese, Catalan, English, Galician, and Basque (in order of the number of documents in the dataset) (Zubiaga et al., 2014, 2016). The organizers provided almost 35,000 Twitter messages, and in addition to the six monolingual tags, supported four additional categories: undetermined, multilingual (i.e. the message contains more than one language, without requiring the system to specify the component languages), ambiguous (i.e. the message is ambiguous between two or more of the six target languages), and other (i.e. the message is in a language other than the six target languages). The second shared task was the PAN lab on authorship profiling 2017 (Rangel et al., 2017b). The PAN lab on authorship profiling is held annually and historically has focused on age, gender,

Year – Title	Reference
2010 – DÉfi Fouille de Texte (DEFT) `https://deft.limsi.fr`	Grouin et al. (2011)
2010 – Australasian Language Technology Workshop `http://www.alta.asn.au`	Baldwin and Lui (2010b)
2014 – Twitter Language Identification Workshop at SEPLN 2014 `http://komunitatea.elhuyar.org/tweetlid/?lang=en_us`	Zubiaga et al. (2014)
2014 – Computational Approaches to Code Switching `http://emnlp2014.org/workshops/CodeSwitch/call.html`	Solorio et al. (2014)
2014 – First DSL Shared Task at VarDial `http://corporavm.uni-koeln.de/vardial/sharedtask.html`	Zampieri et al. (2014)
2015 – Second DSL Shared Task at VarDial `http://ttg.uni-saarland.de/lt4vardial2015/dsl.html`	Zampieri et al. (2015)
2016 – First Arabic Dialect Identification (ADI) at VarDial `http://ttg.uni-saarland.de/vardial2016/dsl2016.html`	Malmasi et al. (2016)
2016 – Third DSL Shared Task at VarDial `http://ttg.uni-saarland.de/vardial2016/dsl2016.html`	Malmasi et al. (2016)
2017 – Second Arabic Dialect Identification (ADI) at VarDial `http://ttg.uni-saarland.de/vardial2017/sharedtask2017.html`	Zampieri et al. (2017)
2017 – Fourth DSL Shared Task at VarDial `http://ttg.uni-saarland.de/vardial2017/sharedtask2017.html`	Zampieri et al. (2017)
2017 – First German Dialect Identification (ADI) at VarDial `http://ttg.uni-saarland.de/vardial2017/sharedtask2017.html`	Zampieri et al. (2017)
2017 – PAN lab on Author Profiling `http://pan.webis.de/clef17/pan17-web/author-profiling.html`	Rangel et al. (2017b)
2017 – Arabic Multi-Genre Broadcast (MGB) Challenge `http://www.mgb-challenge.org/arabic.html`	Ali et al. (2017)

Table 12: List of LI shared tasks.

and personality traits prediction in social media. In 2017, the competition introduced the inclusion of language varieties and dialects of Arabic, English, Spanish, and Portuguese.

More ambitiously, the four editions of the Discriminating between Similar Languages (DSL) (Zampieri et al., 2014, 2015; Malmasi et al., 2016; Zampieri et al., 2017) shared tasks required participants to discriminate between a set of languages in several language groups, each consisting of highly-similar languages or national varieties of that language. The dataset, entitled DSL Corpus Collection ("DSLCC") (Tan et al., 2014), and the languages included are summarized in Table 13. Historically the best-performing systems (Goutte et al., 2014; Lui et al., 2014b; Bestgen, 2017) have approached the task via hierarchical classification, first predicting the language group, then the language within that group.

Language/Variety	v1.0 (2014)	v2.0/2.1 (2015)	v3.0 (2016)	v4.0 (2017)
Bosnian	✓	✓	✓	✓
Croatian	✓	✓	✓	✓
Serbian	✓	✓	✓	✓
Czech	✓	✓		
Slovak	✓	✓		
Indonesian	✓	✓	✓	✓
Malay	✓	✓	✓	✓
Brazilian Portuguese	✓	✓	✓	✓
European Portuguese	✓	✓	✓	✓
Macanese Portuguese		✓		
Argentine Spanish	✓	✓	✓	✓
Castilian Spanish	✓	✓	✓	✓
Mexican Spanish		✓	✓	
Peruvian Spanish				✓
Bulgarian		✓		
Macedonian		✓		
Canadian French			✓	✓
Hexagonal French			✓	✓
American English	✓			
British English	✓			
Persian			✓	
Dari			✓	

Table 13: DSLCC: the languages included in each version of the corpus collection, grouped by language similarity.

8. Application Areas

There are various reasons to investigate LI. Studies in LI approach the task from different perspectives, and with different motivations and application goals in mind. In this section, we briefly summarize what these motivations are, and how their specific needs differ.

The oldest motivation for automatic LI is perhaps in conjunction with translation (Beesley, 1988). Automatic LI is used as a pre-processing step to determine what translation model to apply to an input text, whether it be by routing to a specific human translator or by applying MT. Such a use case is still very common, and can be seen in the Google Chrome web browser (Google, 2019), where a built-in LI module is used to offer MT services to the user when the detected language of the web page being visited differs from the user's language settings.

NLP components such as POS taggers and parsers tend to make a strong assumption that the input text is monolingual in a given language. Similarly to the translation case, LI can play an obvious role in routing documents written in different languages to NLP components tailored to those languages. More subtle is the case of documents with mixed multilingual content, the most commonly-occurring instance of which is foreign inclusion, where a document is predominantly in a single language (e.g. German or Japanese) but

is interspersed with words and phrases (often technical terms) from a language such as English. For example, Alex, Dubey, and Keller (2007) found that around 6% of word tokens in German text sourced from the Internet are English inclusions. In the context of POS tagging, one strategy for dealing with inclusions is to have a dedicated POS for all foreign words, and force the POS tagger to perform both foreign inclusion detection and POS-tag these words in the target language; this is the approach taken in the Penn POS tagset, for example (Marcus, Santorini, & Marcinkiewicz, 1993). An alternative strategy is to have an explicit foreign inclusion detection pre-processor, and some special handling of foreign inclusions. For example, in the context of German parsing, Alex et al. (2007) used foreign inclusion predictions to restrict the set of (German) POS-tags used to form a parse tree, and found that this approach substantially improved parser accuracy.

Another commonly-mentioned use case is for multilingual document storage and retrieval. A document retrieval system (such as, but not limited to, a web search engine) may be required to index documents in multiple languages. In such a setting, it is common to apply LI at two points: (1) to the documents being indexed; and (2) to the queries being executed on the collection. Simple keyword matching techniques can be problematic in text-based document retrieval, because the same word can be valid in multiple languages. A classic example of such words (known as "false friends") includes *Gift*, which in German means "poison". Performing LI on both the document and the query helps to avoid confusion between such terms, by taking advantage of the context in which it appears in order to infer the language. This has resulted in specific work in LI of web pages, as well as search engine queries. Roy, Choudhury, Majumder, and Agarwal (2013) and Sequeira et al. (2015) give overviews of shared tasks specifically concentrating on language labeling of individual search query words. Having said this, in many cases, the search query itself does a sufficiently good job of selecting documents in a particular language, and overt LI is often not performed in mixed multilingual search contexts.

Automatic LI has also been used to facilitate linguistic and other text-based research. Suzuki et al. (2002) report that their motivation for developing a language identifier was "to find out how many web pages are written in a particular language". Automatic LI has been used in constructing web-based corpora. The Crúbadán project (Scannell, 2007) and the Finno-Ugric Languages and the Internet project (Jauhiainen, Jauhiainen, & Lindén, 2015) make use of automated LI techniques to gather linguistic resources for under-resourced languages. Similarly, the Online Database of INterlinear text ("ODIN": Lewis & Xia, 2010) uses automated LI as one of the steps in collecting interlinear glossed text from the web for purposes of linguistic search and bootstrapping NLP tools.

One challenge in collecting linguistic resources from the web is that documents can be multilingual (i.e. contain text in more than one language). This is problematic for standard LI methods, which assume that a document is written in a single language, and has prompted research into segmenting text by language, as well as word-level LI, to enable extraction of linguistic resources from multilingual documents. A number of LI shared tasks discussed in detail in Section 7.3 included data from social media. Examples are the TweetLID shared task on tweet LI held at SEPLN 2014 (Zubiaga et al., 2014, 2016), the data sets used in the first and second shared tasks on LI in code-switched data which were partially taken from Twitter (Solorio et al., 2014; Molina et al., 2016), and the third edition of the DSL shared task which contained two out-of-domain test sets consisting of

tweets (Malmasi et al., 2016). The 5th edition of the PAN at CLEF author profiling task included language variety identification for tweets (Rangel et al., 2017b). There has also been research on identifying the language of private messages between eBay users (Mayer, 2012), presumably as a filtering step prior to more in-depth data analysis.

9. Off-the-Shelf Language Identifiers

An "off-the-shelf" language identifier is software that is distributed with pre-trained models for a number of languages, so that a user is not required to provide training data before using the system. Such a setup is highly attractive to many end-users of automatic LI whose main interest is in utilizing the output of a language identifier rather than implementing and developing the technique. To this end, a number of off-the-shelf language identifiers have been released over time. Many authors have evaluated these off-the-shelf identifiers, including a recent evaluation involving 13 language identifiers which was carried out by Pawelka and Jürgens (2015). In this section, we provide a brief summary of open-source or otherwise free systems that are available, as well as the key characteristics of each system. We have also included dates of when the software has been last updated as of October 2018.

`TextCat` is the most well-known Perl implementation of the out-of-place method, it lists models for 76 languages in its off-the-shelf configuration; the program is not actively maintained (van Noord, 1997). `TextCat` is not the only example of an off-the-shelf implementation of the out-of-place method: other implementations include `libtextcat` with 76 language models (Scheelen, 2003), `JTCL` with 15 languages (Hammerl, 2013), and `mguesser` with 104 models for different language-encoding pairs (Barkov, 2008). The main issue addressed by later implementations is classification speed: `TextCat` is implemented in Perl and is not optimized for speed, whereas implementations such as `libtextcat` and `mguesser` have been specifically written to be fast and efficient. `whatlang-rs` uses an algorithm based on character trigrams and refers the user to the Cavnar and Trenkle (1994) article. It comes pre-trained with 83 languages (Popatov, 2016).

`ChromeCLD` is the language identifier embedded in the Google Chrome web browser (Sites, 2013). It uses a NB classifier, and script-specific classification strategies. `ChromeCLD` assumes that all the input is in UTF-8, and assigns the responsibility of encoding detection and transcoding to the user. `ChromeCLD` uses Unicode information to determine the script of the input. `ChromeCLD` also implements a number of pre-processing heuristics to help boost performance on its target domain (web pages), such as stripping character sequences like `.jpg`. The standard implementation supports 83 languages, and an extended model is also available, that supports 160 languages.

`LangDetect` is a Java library that implements a language identifier based on a NB classifier trained over character n-grams. The software comes with pre-trained models for 53 languages, using data from Wikipedia (Nakatani, 2010). `LangDetect` makes use of a range of normalization heuristics to improve the performance on particular languages, including: (1) clustering of Chinese/Japanese/Korean characters to reduce sparseness; (2) removal of "language-independent" characters, and other text normalization; and (3) normalization of Arabic characters.

`langid.py` is a Python implementation of the method described by Lui and Baldwin (2011), which exploits training data for the same language across multiple different sources

of text to identify sequences of characters that are strongly predictive of a given language, regardless of the source of the text. This feature set is combined with a NB classifier, and is distributed with a pre-trained model for 97 languages prepared using data from 5 different text sources (Lui, 2011). Lui and Baldwin (2012) provide an empirical comparison of `langid.py` to `TextCat`, `LangDetect` and `ChromeCLD` and find that it compares favorably both in terms of accuracy and classification speed. There are also implementations of the classifier component (but not the training portion) of `langid.py` in Java (Weiss, 2013), C (Lui, 2014b), and JavaScript (Lui, 2014c).

`whatlang` (Brown, 2013) uses a vector-space model with per-feature weighting on character n-gram sequences. One particular feature of `whatlang` is that it uses discriminative training in selecting features, i.e. it specifically makes use of features that are strong evidence *against* a particular language, which is generally not captured by NB models. Another feature of `whatlang` is that it uses inter-string smoothing to exploit sentence-level locality in making language predictions, under the assumption that adjacent sentences are likely to be in the same language. Brown (2013) reports that this substantially improves the accuracy of the identifier. Another distinguishing feature of `whatlang` is that it comes pre-trained with data for 1400 languages, which is the highest number by a large margin of any off-the-shelf system (Brown, 2014a).

`whatthelang` is a recent language identifier written in Python, which utilizes the FastText NN-based text classification algorithm. It supports 176 languages (Sangeeth, 2017).

`YALI` implements an off-the-shelf classifier trained using Wikipedia data, covering 122 languages (Majliš, 2012a). Although not described as such, the actual classification algorithm used is a linear model, and is thus closely related to both NB and a cosine-based vector space model.

In addition to the above-mentioned general-purpose language identifiers, there have also been efforts to produce pre-trained language identifiers targeted specifically at Twitter messages. `LDIG` is a Twitter-specific LI tool with built-in models for 19 languages (Nakatani, 2011). It uses a document representation based on tries (Okanohara & Tsujii, 2009). The algorithm is a LR classifier using all possible substrings of the data, which is important to maximize the available information from the relatively short Twitter messages.

Lui and Baldwin (2014) provides a comparison of 8 off-the-shelf language identifiers applied without re-training to Twitter messages. One issue they report is that comparing the accuracy of off-the-shelf systems is difficult because of the different subset of languages supported by each system, which may also not fully cover the languages present in the target data. The authors choose to compare accuracy over the full set of languages, arguing that this best reflects the likely use-case of applying an off-the-shelf LI system to new data. They find that the best individual systems are `ChromeCLD`, `langid.py` and `LangDetect`, but that slightly higher accuracy can be attained by a simple voting-based ensemble classifier involving these three systems.

In addition to this, commercial or other closed-source language identifiers and language identifier services exist, of which we name a few. The Polyglot 3000 (Likasoft, 2015) and Lextek Language Identifier (Lextek, 2019) are standalone language identifiers for Windows.

10. Research Directions and Open Issues in LI

Several papers have catalogued open issues in LI (Sibun & Reynar, 1996; Xia et al., 2010; Hughes et al., 2006; da Silva & Lopes, 2006b; Baldwin & Lui, 2010a; Botha & Barnard, 2012; Malmasi et al., 2016). Some of the issues, such as text representation (Section 5) and choice of algorithm (Section 6), have already been covered in detail in this survey. In this section, we synthesize the remaining issues into a single section, and also add new issues that have not been discussed in previous work. For each issue, we review related work and suggest promising directions for future work.

10.1 Text Preprocessing

Text preprocessing (also known as normalization) is an umbrella term for techniques where an automatic transformation is applied to text before it is presented to a classifier. The aim of such a process is to eliminate sources of variation that are expected to be confounding factors with respect to the target task. Text preprocessing is slightly different from data cleaning, as data cleaning is a transformation applied only to training data, whereas normalization is applied to both training and test data. Hughes et al. (2006) raise text preprocessing as an outstanding issue in LI, arguing that its effects on the task have not been sufficiently investigated. In this section, we summarize the normalization strategies that have been proposed in the LI literature.

Case folding is the elimination of capitalization, replacing characters in a text with either their lower-case or upper-case forms. Basic approaches generally map between `[a-z]` and `[A-Z]` in the ASCII encoding, but this approach is insufficient for extended Latin encodings, where diacritics must also be appropriately handled. A resource that makes this possible is the Unicode Character Database (UCD)(Unicode, 2019) which defines uppercase, lowercase and titlecase properties for each character, enabling automatic case folding for documents in a Unicode encoding such as UTF-8.

Range compression is the grouping of a range of characters into a single logical set for counting purposes, and is a technique that is commonly used to deal with the sparsity that results from character sets for ideographic languages, such as Chinese, that may have thousands of unique "characters", each of which is observed with relatively low frequency. Simões, Almeida, and Byers (2014) use such a technique where all characters in a given range are mapped into a single "bucket", and the frequency of items in each bucket is used as a feature to represent the document. Byte-level representations of encodings that use multi-byte sequences to represent codepoints achieve a similar effect by "splitting" codepoints. In encodings such as UTF-8, the codepoints used by a single language are usually grouped together in "code planes", where each codepoint in a given code plane shares the same upper byte. Thus, even though the distribution over codepoints may be quite sparse, when the byte-level representation uses byte sequences that are shorter than the multi-byte sequence of a codepoint, the shared upper byte will be predictive of specific languages.

Cleaning may also be applied, where heuristic rules are used to remove some data that is perceived to hinder the accuracy of the language identifier. For example, Suzuki et al. (2002) identify HTML entities as a candidate for removal in document cleaning, on the basis that classifiers trained on data which does not include such entities may drop in accuracy when applied to raw HTML documents. `ChromeCLD` includes heuristics such as expanding

HTML entities, deleting digits and punctuation, and removing SGML-like tags. Similarly, `LangDetect` also removes "language-independent characters" such as numbers, symbols, URLs, and email addresses. It also removes words that are all-capitals and tries to remove other acronyms and proper names using heuristics.

In the domain of Twitter messages, Tromp and Pechenizkiy (2011) remove links, usernames, smilies, and hashtags (a Twitter-specific "tagging" feature), arguing that these entities are language independent and thus should not feature in the model. Xafopoulos et al. (2004) address LI of web pages, and report removing HTML formatting, and applying stopping using a small stopword list. Takçı and Ekinci (2012) carry out LI experiments on the ECI multilingual corpus and report removing punctuation, space characters, and digits.

The idea of preprocessing text to eliminate domain-specific "noise" is closely related to the idea of learning domain-independent characteristics of a language (Lui & Baldwin, 2011). One difference is that normalization is normally heuristic-driven, where a manually-specified set of rules is used to eliminate unwanted elements of the text, whereas domain-independent text representations are data-driven, where text from different sources is used to identify the characteristics that a language shares between different sources. Both approaches share conceptual similarities with problems such as content extraction for web pages. In essence, the aim is to isolate the components of the text that actually represent language, and suppress the components that carry other information. One application is the language-aware extraction of text strings embedded in binary files, which has been shown to perform better than conventional heuristic approaches (Brown, 2012). Future work in this area could focus specifically on the application of language-aware techniques to content extraction, using models of language to segment documents into textual and non-textual components. Such methods could also be used to iteratively improve LI itself by improving the quality of training data.

10.2 Orthography and Transliteration

LI is further complicated when we consider that some languages can be written in different orthographies (e.g. Bosnian and Serbian can be written in both Latin and Cyrillic script). Transliteration is another phenomenon that has a similar effect, whereby phonetic transcriptions in another script are produced for particular languages. These transcriptions can either be standardized and officially sanctioned, such as the use of *Hanyu Pinyin* for Chinese, or may also emerge irregularly and organically as in the case of *arabizi* for Arabic (Yaghan, 2008). Hughes et al. (2006) identify variation in the encodings and scripts used by a given language as an open issue in LI, pointing out that early work tended to focus on languages written using a romanized script, and suggesting that dealing with issues of encoding and orthography adds substantial complexity to the task. Suzuki et al. (2002) discuss the relative difficulties of discriminating between languages that vary in any combination of encoding, script and language family, and give examples of pairs of languages that fall into each category.

LI across orthographies and transliteration is an area that has not received much attention in work to date, but presents unique and interesting challenges that are suitable targets for future research. An interesting and unexplored question is whether it is possible to detect that documents in different encodings or scripts are written in the same

language, or what language a text is transliterated from, without any a-priori knowledge of the encoding or scripts used. One possible approach to this could be to take advantage of standard orderings of alphabets in a language – the pattern of differences between adjacent characters should be consistent across encodings, though whether this is characteristic of any given language requires exploration.

10.3 Supporting Low-Resource Languages

Hughes et al. (2006) paint a fairly bleak picture of the support for low-resource languages in automatic LI. This is supported by the arguments of Xia et al. (2010) who detail specific issues in building hugely multilingual datasets. Abney and Bird (2010) also specifically called for research into automatic LI for low-density languages. Ethnologue (Simons & Fennig, 2017) lists a total of 7099 languages. Xia et al. (2010) describe the Ethnologue in more detail, and discuss the role that LI plays in other aspects of supporting minority languages, including detecting and cataloging resources. The problem is circular: LI methods are typically supervised, and need training data for each language to be covered, but the most efficient way to recover such data is through LI methods.

A number of projects are ongoing with the specific aim of gathering linguistic data from the web, targeting as broad a set of languages as possible. One such project is the aforementioned ODIN (Xia et al., 2009; Lewis & Xia, 2010), which aims to collect parallel snippets of text from Linguistics articles published on the web. ODIN specifically targets articles containing Interlinear Glossed Text (IGT), a semi-structured format for presenting text and a corresponding gloss that is commonly used in Linguistics.

Other projects that exist with the aim of creating text corpora for under-resourced languages by crawling the web are the Crúbadán project (Scannell, 2007) and SeedLing (Emerson, Tan, Fertmann, Palmer, & Regneri, 2014). The Crúbadán crawler uses seed data in a target language to generate word lists that in turn are used as queries for a search engine. The returned documents are then compared with the seed resource via an automatic language identifier, which is used to eliminate false positives. Scannell (2007) reports that corpora for over 400 languages have been built using this method. The SeedLing project crawls texts from several web sources which has resulted in a total of 1451 languages from 105 language families. According to the authors, this represents 19% of the world's languages.

Much recent work on multilingual documents (Section 10.6) has been done with support for minority languages as a key goal. One of the common problems with gathering linguistic data from the web is that the data in the target language is often embedded in a document containing data in another language. This has spurred recent developments in text segmentation by language and word-level LI. Lui et al. (2014a) present a method to detect documents that contain text in more than one language and identify the languages present with their relative proportions in the document. The method is evaluated on real-world data from a web crawl targeted to collect documents for specific low-density languages.

LI for low-resource languages is a promising area for future work. One of the key questions that has not been clearly answered is how much data is needed to accurately model a language for purposes of LI. Work to date suggests that there may not be a simple answer to this question as accuracy varies according to the number and variety of languages

modeled (Baldwin & Lui, 2010a), as well as the diversity of data available to model a specific language (Lui & Baldwin, 2011).

10.4 Number of Languages

Early research in LI tended to focus on a very limited number of languages (sometimes as few as 2). This situation has improved somewhat with many current off-the-shelf language identifiers supporting on the order of 50–100 languages (Section 9). The standout in this regard is Brown (2014b), supporting 1311 languages in its default configuration. However, evaluation of the identifier of Brown (2013) on a different domain found that the system suffered in terms of accuracy because it detected many languages that were not present in the test data (Lui & Baldwin, 2014).

Lewis and Xia (2010) describe the construction of web crawlers specifically targeting IGT, as well as the identification of the languages represented in the IGT snippets. LI for thousands of languages from very small quantities of text is one of the issues that they have had to tackle. They list four specific challenges for LI in ODIN: (1) the large number of languages; (2) "unseen" languages that appear in the test data but not in training data; (3) short target sentences; and (4) (sometimes inconsistent) transliteration into Latin text. Their solution to this task is to take advantage of a domain-specific feature: they assume that the name of the language that they are extracting must appear in the document containing the IGT, and hence treat this as a co-reference resolution problem. They report that this approach significantly outperforms the text-based LI approach in this particular problem setting.

An interesting area to explore is the trade-off between the number of languages supported and the accuracy per-language. From existing results it is not clear if it is possible to continue increasing the number of languages supported without adversely affecting the average accuracy, but it would be useful to quantify if this is actually the case across a broad range of text sources. Table 14 lists the articles where the LI with more than 30 languages has been investigated.

10.5 "Unseen" Languages and Unsupervised LI

"Unseen" languages are languages that we do not have training data for but may nonetheless be encountered by a LI system when applied to real-world data. Dealing with languages for which we do not have training data has been identified as an issue by Hughes et al. (2006) and has also been mentioned by Xia et al. (2009) as a specific challenge in harvesting linguistic data from the web. Elfardy and Diab (2012) use an unlabeled training set with a labeled evaluation set for token-level code-switching identification between Modern Standard Arabic (MSA) and dialectal Arabic. They utilize existing dictionaries and also a morphological analyzer for MSA, so the system is supported by extensive external knowledge sources. The possibility to use unannotated training material is nonetheless a very useful feature.

Some authors have attempted to tackle the unseen language problem through attempts at unsupervised labeling of text by language. Mather (1998) uses an unsupervised clustering algorithm to separate a multilingual corpus into groups corresponding to languages. She uses singular value decomposition (SVD) to first identify the words that discriminate between documents and then to separate the terms into highly correlating groups. The

Reference	# Lang	**Reference**	# Lang
Brown (2014b)	1366	Brown (2013)	1100
Brown (2012)	923	Xia et al. (2009)	c. 600
Rodrigues (2012)	372	King and Dehdari (2008)	300
Jauhiainen, Lindén, and Jauhiainen (2015b)	285	Jauhiainen et al. (2017b)	285
Vatanen et al. (2010)	281	Yamaguchi and Tanaka-Ishii (2012)	>200
Cazamias et al. (2015)	200	Chew, Mikami, and Nagano (2011)	182
Lui (2014a)	143	Kocmi and Bojar (2017)	136
Majliš (2011)	122	Jauhiainen (2010)	103
Majliš (2012b)	90	Lui and Baldwin (2011)	89
Baldwin and Lui (2010b)	74	Chew, Mikami, Marasinghe, and Nandasara (2009)	68
Baldwin and Lui (2010a)	67	Lui and Baldwin (2012)	67
Lui and Baldwin (2014)	65	Goldszmidt et al. (2013)	52
Chen and Maison (2003)	48	Lui et al. (2014a)	44
Singh (2006)	39	Cowie et al. (1999)	34
Ludovik and Zacharski (1999)	34	Hammarström (2007)	32
Abainia, Ouamour, and Sayoud (2014)	32	King and Abney (2013)	31

Table 14: Empirical evaluations with more than 30 languages.

documents grouped together by these discriminating terms are merged and the process is repeated until the wanted number of groups (corresponding to languages) is reached. Biemann and Teresniak (2005) also present an approach to the unseen language problem, building graphs of co-occurrences of words in sentences, and then partitioning the graph using a custom graph-clustering algorithm which labels each word in the cluster with a single label. The number of labels is initialized to be the same as the number of words, and decreases as the algorithm is recursively applied. After a small number of iterations (the authors report 20), the labels become relatively stable and can be interpreted as cluster labels. Smaller clusters are then discarded, and the remaining clusters are interpreted as groups of words for each language. Shiells and Pham (2010) compared the Chinese Whispers algorithm of Biemann and Teresniak (2005) and Graclus clustering on unsupervised Tweet LI. They conclude that Chinese Whispers is better suited to LI. Selamat and Ng (2008) used Fuzzy ART NNs for unsupervised language clustering for documents in Arabic, Persian, and Urdu. In Fuzzy ART, the clusters are also dynamically updated during the identification process.

Amine, Elberrichi, and Simonet (2010) also tackle the unseen language problem through clustering. They use a character n-gram representation for text, and a clustering algorithm that consists of an initial k-means phase, followed by particle-swarm optimization. This produces a large number of small clusters, which are then labeled by language through a separate step. Wan (2016) used co-occurrences of words with k-means clustering in word-level unsupervised LI. They used a Dirichlet process Gaussian mixture model ("DPGMM"), a non-parametric variant of a GMM, to automatically determine the number of clusters, and manually labeled the language of each cluster. Poulston et al. (2017) also used k-means clustering, and Alfter (2015) used the x-means clustering algorithm in a custom framework. Lin et al. (2014) utilized unlabeled data to improve their LI system by using a CRF autoencoder, unsupervised word embeddings, and word lists.

A different partial solution to the issue of unseen languages is to design the classifier to be able to output "unknown" as a prediction for language. This helps to alleviate one of the problems commonly associated with the presence of unseen languages – classifiers without an "unknown" facility are forced to pick a language for each document, and in the case of unseen languages, the choice may be arbitrary and unpredictable (Biemann & Teresniak, 2005). When LI is used for filtering purposes, i.e. to select documents in a single language, this mislabeling can introduce substantial noise into the data extracted; furthermore, it does not matter what or how many unseen languages there are, as long as they are consistently rejected. Therefore the "unknown" output provides an adequate solution to the unseen language problem for purposes of filtering.

The easiest way to implement unknown language detection is through thresholding. Most systems internally compute a score for each language for an unknown text, so thresholding can be applied either with a global threshold (Cowie et al., 1999), a per-language threshold (Suzuki et al., 2002), or by comparing the score for the top-scoring N-languages. The problem of unseen languages and open-set recognition was also considered by Malmasi and Dras (2015b), Zampieri et al. (2015), and Malmasi (2017). Malmasi (2017) experiments with one-class classification ("OCC") and reaches an F-score on 98.9 using OC-SVMs (SVMs trained only with data from one language) to discriminate between 10 languages.

Another possible method for unknown language detection that has not been explored extensively in the literature, is the use of non-parametric mixture models based on Hierarchical Dirichlet Processes ("HDP"). Such models have been successful in topic modeling, where an outstanding issue with the popular LDA model is the need to specify the number of topics in advance. Lui et al. (2014a) introduced an approach to detecting multilingual documents that uses a model very similar to LDA, where languages are analogous to topics in the LDA model. Using a similar analogy, an HDP-based model may be able to detect documents that are written in a language that is not currently modeled by the system. Voss et al. (2014) used LDA to cluster unannotated tweets. Recently Zhang, Clark, Wang, and Li (2016) used LDA in unsupervised sentence-level LI. They manually identified the languages of the topics created with LDA. If there were more topics than languages then the topics in the same language were merged.

Filtering, a task that we mentioned earlier in this section, is a very common application of LI, and it is therefore surprising that there is little research on filtering for specific languages. Filtering is a limit case of LI with unseen languages, where all languages but one can be considered unknown. Future work could examine how useful different types of negative evidence are for filtering – if we want to detect English documents, e.g., are there empirical advantages in having distinct models of Italian and German (even if we don't care about the distinction between the two languages), or can we group them all together in a single "negative" class? Are we better off including as many languages as possible in the negative class, or can we safely exclude some?

10.6 Multilingual Documents

Multilingual documents are documents that contain text in more than one language. In constructing the hrWac corpus (Ljubešić & Klubička, 2014), Stupar et al. (2011) found that 4% of the documents they collected contained text in more than one language. Martins and

Silva (2005) report that web pages in many languages contain formulaic strings in English that do not actually contribute to the content of the page, but may nonetheless confound attempts to identify multilingual documents. Recent research has investigated how to make use of multilingual documents from sources such as web crawls (King & Abney, 2013), forum posts (Nguyen & Dogruöz, 2013), and microblog messages (Ling, Xiang, Dyer, Black, & Trancoso, 2013). However, most LI methods assume that a document contains text from a single language, and so are not directly applicable to multilingual documents.

Handling of multilingual documents has been named as an open research question (Hughes et al., 2006). Most NLP techniques presuppose monolingual input data, so inclusion of data in foreign languages introduces noise, and can degrade the performance of NLP systems. Automatic detection of multilingual documents can be used as a prefiltering step to improve the quality of input data. Detecting multilingual documents is also important for acquiring linguistic data from the web, and has applications in mining bilingual texts for statistical MT from online resources (Ling et al., 2013), or to study code-switching phenomena in online communications. There has also been interest in extracting text resources for low-density languages from multilingual web pages containing both the low-density language and another language such as English.

The need to handle multilingual documents has prompted researchers to revisit the granularity of LI. Many researchers consider document-level LI to be relatively easy, and that sentence-level and word-level LI are more suitable targets for further research. However, word-level and sentence-level tokenization are not language-independent tasks, and for some languages are substantially harder than others (Peng, Feng, & McCallum, 2004).

`Linguini` (Prager, 1999) is a language identifier that supports identification of multilingual documents. The system is based on a vector space model using cosine similarity. LI for multilingual documents is performed through the use of *virtual mixed languages*. Prager (1999) shows how to construct vectors representative of particular combinations of languages independent of the relative proportions, and proposes a method for choosing combinations of languages to consider for any given document. One weakness of this approach is that for exhaustive coverage, this method is factorial in the number of languages, and as such intractable for a large set of languages. Furthermore, calculating the parameters for the virtual mixed languages becomes infeasibly complex for mixtures of more than 3 languages.

As mentioned previously, Lui et al. (2014a) propose an LDA-inspired LI method for multilingual documents that is able to identify that a document is multilingual, identify the languages present and estimate the relative proportions of the document written in each language. To remove the need to specify the number of topics (or in this case, languages) in advance, Lui et al. (2014a) use a greedy heuristic that attempts to find the subset of languages that maximizes the posterior probability of a target document. One advantage of this approach is that it is not constrained to 3-language combinations like the method of Prager (1999). Language set identification has also been considered by Suzuki et al. (2002), Jauhiainen et al. (2015b), and Pla and Hurtado (2015, 2017).

To encourage further research on LI for multilingual documents, in the aforementioned shared task hosted by the Australiasian Language Technology Workshop 2010, discussed in Section 7.3, participants were required to predict the language(s) present in a held-out test set containing monolingual and bilingual documents (Baldwin & Lui, 2010b). The dataset

was prepared using data from Wikipedia, and bilingual documents were produced using a segment from an article in one language and a segment from the equivalent article in another language. Equivalence between articles was determined using the cross-language links embedded within each Wikipedia article.[9] The winning entry (Tran, Nguyen, & Kieu, 2010) first built monolingual models from multilingual training data, and then applied them to a chunked version of the test data, making the final prediction a function of the prediction over chunks.

Another approach to handling multilingual documents is to attempt to segment them into contiguous monolingual segments. In addition to identifying the languages present, this requires identifying the locations of boundaries in the text which mark the transition from one language to another. Several methods for supervised language segmentation have been proposed. Cowie et al. (1999) generalized a LI algorithm for monolingual documents by adding a dynamic programming algorithm based on a simple Markov model of multilingual documents. More recently, multilingual LI algorithms have also been presented by Jhamtani et al. (2014), Minocha and Tyers (2014), Pethö and Mózes (2014), Ullman (2014), and King et al. (2015).

10.7 Short Texts

LI of short strings is known to be challenging for existing LI techniques. Mandl et al. (2006) tested four different classification methods, and found that all have substantially lower accuracy when applied to texts of 25 characters compared with texts of 125 characters. These findings were later strengthened, for example, by Vatanen et al. (2010) and Jauhiainen et al. (2017b).

Hammarström (2007) describes a method specifically targeted at short texts that augments a dictionary with an affix table, which was tested over synthetic data derived from a parallel bible corpus. Vatanen et al. (2010) focus on messages of 5–21 characters, using n-gram language models over data drawn the from Universal Declaration of Human Rights (UDHR). We would expect that generic methods for LI of short texts should be effective in any domain where short texts are found, such as search engine queries or microblog messages. However, Hammarström (2007) and Vatanen et al. (2010) both only test their systems in a single domain: bible texts in the former case, and texts from the UDHR in the latter case. Other research has shown that LI results do not trivially generalize across domains (Baldwin & Lui, 2010a), and found that LI in UDHR documents is relatively easy (Yamaguchi & Tanaka-Ishii, 2012). For both bible and UDHR data, we expect that the linguistic content is relatively grammatical and well-formed, an expectation that does not carry across to domains such as search engine queries and microblogs. Another "short text" domain where LI has been studied is LI of proper names. Häkkinen and Tian (2001) identify this as an issue. Konstantopoulos (2007) found that LI of names is more accurate than LI of generic words of equivalent length.

Bergsma et al. (2012) raise an important criticism of LI work on Twitter messages to date: only a small number of European languages has been considered. Bergsma et al. (2012) expand the scope of LI for Twitter, covering nine languages across Cyrillic, Arabic

9. Note that such articles are not necessarily direct translations but rather articles about the same topic written in different languages.

and Devanagari scripts. Lui and Baldwin (2014) expand the evaluation further, introducing a dataset of language-labeled Twitter messages across 65 languages constructed using a semi-automatic method that leverages user identity to avoid inducing a bias in the evaluation set towards messages that existing systems are able to identify correctly. Lui and Baldwin (2014) also test a 1300-language model based on Brown (2013), but find that it performs relatively poorly in the target domain due to a tendency to over-predict low-resource languages.

Work has also been done on LI of single words in a document, where the task is to label each word in the document with a specific language. Work to date in this area has assumed that word tokenization can be carried out on the basis of whitespace. Singh and Gorla (2007) explore word-level LI in the context of segmenting a multilingual document into monolingual segments. Other work has assumed that the languages present in the document are known in advance.

Conditional random fields ("CRFs": Lafferty et al., 2001) are a sequence labeling method most often used in LI for labeling the language of individual words in a multilingual text. CRFs can be thought of as a finite state model with probabilistic transition probabilities optimised over pre-defined cliques. They can use any observations made from the test document as features, including language labels given by monolingual language identifiers for words. King and Abney (2013) used a CRF trained with generalized expectation criteria, and found it to be the most accurate of all methods tested (NB, LR, HMM, CRF) at word-level LI. King and Abney (2013) introduce a technique to estimate the parameters using only monolingual data, an important consideration as there is no readily-available collection of manually-labeled multilingual documents with word-level annotations. Nguyen and Dogruöz (2013) present a two-pass approach to processing Turkish-Dutch bilingual documents, where the first pass labels each word independently and the second pass uses the local context of a word to further refine the predictions. Nguyen and Dogruöz (2013) achieved 97,6% accuracy on distinguishing between the two languages using a linear-chain CRF. Clematide and Makarov (2017) are the only ones so far to use a CRF for LI of monolingual texts. With a CRF, they attained a higher F-score in German dialect identification than NB or an ensemble consisting of NB, CRF, and SVM. Lately CRFs were also used for LI by Dongen (2017) and Samih (2017). Giwa and Davel (2013) investigate LI of individual words in the context of code-switching. They find that smoothing of n-gram models substantially improves accuracy of a language identifier based on a NB classifier when applied to individual words.

10.8 Similar Languages, Language Varieties, and Dialects

While one line of research into LI has focused on pushing the boundaries of how many languages are supported simultaneously by a single system (Xia et al., 2010; Brown, 2012, 2013), another has taken a complementary path and focused on LI in groups of similar languages. Research in this area typically does not make a distinction between languages, varieties and dialects, because such terminological differences tend to be politically rather than linguistically motivated (Clyne, 1992; Xia et al., 2010; Zampieri & Gebre, 2012), and from an NLP perspective the challenges faced are very similar.

LI for closely-related languages, language varieties, and dialects has been studied for Malay–Indonesian (Ranaivo-Malançon, 2006), Indian languages (Murthy & Kumar, 2006), South Slavic languages (Ljubešić et al., 2007; Tiedemann & Ljubešić, 2012; Ljubešić & Kranjcić, 2014, 2015), Serbo-Croatian dialects (Zecevic & Vujicic-Stankovic, 2013), English varieties (Lui & Cook, 2013; Simaki et al., 2017), Dutch–Flemish (van der Lee & Bosch, 2017), Dutch dialects (including a temporal dimension) (Trieschnigg, Hiemstra, Theune, de Jong, & Meder, 2012), German Dialects (Hollenstein & Aepli, 2015) Mainland–Singaporean–Taiwanese Chinese (Huang & Lee, 2008), Portuguese varieties (Zampieri & Gebre, 2012; Zampieri et al., 2016), Spanish varieties (Zampieri et al., 2013; Maier & Gómez-Rodríguez, 2014), French varieties (Mokhov, 2010b, 2010a; Diwersy, Evert, & Neumann, 2014), languages of the Iberian Peninsula (Zubiaga et al., 2014), Romanian dialects (Ciobanu & Dinu, 2016), and Arabic dialects (Elfardy & Diab, 2013; Zaidan & Callison-Burch, 2014; Tillmann, Al-Onaizan, & Mansour, 2014; Sadat et al., 2014a; Wray, 2018), the last of which we discuss in more detail in this section. As to off-the-shelf tools which can identify closely-related languages, Zampieri and Gebre (2014) released a LI system trained to identify 27 languages, including 10 language varieties. Closely-related languages, language varieties, and dialects have also been the focus of a number of shared tasks in recent years as discussed in Section 7.3.

Similar languages are a known problem for existing language identifiers (Ranaivo-Malançon, 2006; Zampieri, 2013). Suzuki et al. (2002) identify language pairs from the same language family that also share a common script and the same encoding, as the most difficult to discriminate. Tiedemann and Ljubešić (2012) report that `TextCat` achieves only 45% accuracy when trained and tested on 3-way Bosnian/Serbian/Croatian dataset. Lui and Cook (2013) found that LI methods are not competitive with conventional word-based document categorization methods in distinguishing between national varieties of English. Ranaivo-Malançon (2006) reports that a character trigram model is able to distinguish Malay/Indonesian from English, French, German, and Dutch, but handcrafted rules are needed to distinguish between Malay and Indonesian. One kind of rule is the use of "exclusive words" that are known to occur in only one of the languages. A similar idea is used by Tiedemann and Ljubešić (2012), in automatically learning a "blacklist" of words that have a strong negative correlation with a language – i.e. their presence implies that the text is *not* written in a particular language. In doing so, they achieve an overall accuracy of 98%, far surpassing the 45% of `TextCat`. Brown (2013) also adopts such "discriminative training" to make use of negative evidence in LI.

Zampieri (2013) observed that general-purpose approaches to LI typically use a character n-gram representation of text, but successful approaches for closely-related languages, varieties, and dialects seem to favor a word-based representation or higher-order n-grams (e.g. 4-grams, 5-grams, and even 6-grams) that often cover whole words (Huang & Lee, 2008; Tiedemann & Ljubešić, 2012; Lui & Cook, 2013; Goutte et al., 2016). The study compared character n-grams with word-based representations for LI over varieties of Spanish, Portuguese and French, and found that word-level models performed better for varieties of Spanish, but character n-gram models perform better in the case of Portuguese and French.

To train accurate and robust LI systems that discriminate between language varieties or similar languages, models should ideally be able to capture not only lexical but more abstract systemic differences between languages. One way to achieve this, is by using

features that use de-lexicalized text representations (e.g. by substituting named entities or content words by placeholders), or at a higher level of abstraction, using POS-tags or other morphosyntactic information (Zampieri et al., 2013; Lui et al., 2014b; Bestgen, 2017), or even adversarial machine learning to modify the learned representations to remove such artefacts (Li et al., 2018). Finally, an interesting research direction could be to combine work on closely-related languages with the analysis of regional or dialectal differences in language use (Peirsman, Geeraerts, & Speelman, 2010; Anstein, 2013; Doyle, 2014; Diwersy et al., 2014).

In recent years, there has been a significant increase of interest in the computational processing of Arabic. This is evidenced by a number of research papers in several NLP tasks and applications including the identification/discrimination of Arabic dialects (Elfardy & Diab, 2013; Zaidan & Callison-Burch, 2014). Arabic is particularly interesting for researchers interested in language variation due to the fact that the language is often in a diaglossic situation, in which the standard form (Modern Standard Arabic or "MSA") coexists with several regional dialects which are used in everyday communication.

Among the studies published on the topic of Arabic LI, Elfardy and Diab (2013) proposed a supervised approach to distinguish between MSA and Egyptian Arabic at the sentence level, and achieved up to 85.5% accuracy over an Arabic online commentary dataset (Zaidan & Callison-Burch, 2011). Tillmann et al. (2014) achieved higher results over the same dataset using a linear-kernel SVM classifier.

Zaidan and Callison-Burch (2014) compiled a dataset containing MSA, Egyptian Arabic, Gulf Arabic and Levantine Arabic, and used it to investigate three classification tasks: (1) MSA and dialectal Arabic; (2) four-way classification – MSA, Egyptian Arabic, Gulf Arabic, and Levantine Arabic; and (3) three-way classification – Egyptian Arabic, Gulf Arabic, and Levantine Arabic.

Salloum, Elfardy, Alamir-Salloum, Habash, and Diab (2014) explores the use of sentence-level Arabic dialect identification as a pre-processor for MT, in customizing the selection of the MT model used to translate a given sentence to the dialect it uses. In performing dialect-specific MT, the authors achieve an improvement of 1.0% BLEU score compared with a baseline system which does not differentiate between Arabic dialects.

Finally, in addition to the above-mentioned dataset of Zaidan and Callison-Burch (2011), there are a number of notable multi-dialect corpora of Arabic: a multi-dialect corpus of broadcast speeches used in the ADI shared task (Ali et al., 2016); a multi-dialect corpus of (informal) written Arabic containing newspaper comments and Twitter data (Cotterell & Callison-Burch, 2014); a parallel corpus of 2,000 sentences in MSA, Egyptian Arabic, Tunisian Arabic, Jordanian Arabic, Palestinian Arabic, and Syrian Arabic, in addition to English (Bouamor, Habash, & Oflazer, 2014); a corpus of sentences in 18 Arabic dialects (corresponding to 18 different Arabic-speaking countries) based on data manually sourced from web forums (Sadat et al., 2014a); and finally two recently compiled multi-dialect corpora containing microblog posts from Twitter (Elgabou & Kazakov, 2017; Alshutayri & Atwell, 2017).

While not specifically targeted at identifying language varieties, Jurgens et al. (2017) made the critical observation that when naively trained, LI systems tend to perform most poorly over language varieties from the lowest socio-economic demographics (focusing particularly on the case of English), as they tend to be most under-represented in training

corpora. If, as a research community, we are interested in the social equitability of our systems, it is critical that we develop datasets that are truly representative of the global population, to better quantify and remove this effect. To this end, Jurgens et al. (2017) detail a method for constructing a more representative dataset, and demonstrate the impact of training on such a dataset in terms of alleviating socio-economic bias.

10.9 Domain-specific LI

One approach to LI is to build a generic language identifier that aims to correctly identify the language of a text without any information about the source of the text. Some work has specifically targeted LI across multiple domains, learning characteristics of languages that are consistent between different sources of text (Lui & Baldwin, 2011). However, there are often domain-specific features that are useful for identifying the language of a text. In this survey, our primary focus has been on LI of digitally-encoded text, using only the text itself as evidence on which to base the prediction of the language. Within a text, there can sometimes be domain-specific peculiarities that can be used for LI. For example, Mayer (2012) investigates LI of user-to-user messages in the eBay e-commerce portal. He finds that using only the first two and last two words of a message is sufficient for identifying the language of a message.

11. Conclusions

This article has presented a comprehensive survey on language identification of digitally-encoded text. We have shown that LI is a rich, complex, and multi-faceted problem that has engaged a wide variety of research communities. LI accuracy is critical as it is often the first step in longer text processing pipelines, so errors made in LI will propagate and degrade the performance of later stages. Under controlled conditions, such as limiting the number of languages to a small set of Western European languages and using long, grammatical, and structured text such as government documents as training data, it is possible to achieve near-perfect accuracy. This led many researchers to consider LI a solved problem, as argued by McNamee (2005). However, LI becomes much harder when taking into account the peculiarities of real-world data, such as very short documents (e.g. search engine queries), non-linguistic "noise" (e.g. HTML markup), non-standard use of language (e.g. as seen in social media data), and mixed-language documents (e.g. forum posts in multilingual web forums).

Modern approaches to LI are generally data-driven and are based on comparing new documents with models of each target language learned from data. The types of models as well as the sources of training data used in the literature are diverse, and work to date has not compared and evaluated these in a systematic manner, making it difficult to draw broader conclusions about what the "best" method for LI actually is.

Existing work on LI serves to illustrate that the scope and depth of the problem are much greater than they may first appear. In Section 10, we discussed open issues in LI, identifying the key challenges, and outlining opportunities for future research. Far from being a solved problem, aspects of LI make it an archetypal learning task with subtleties that could be tackled by future work on supervised learning, representation learning, multi-task learning, domain adaptation, multi-label classification and other subfields of machine

learning. We hope that this paper can serve as a reference point for future work in the area, both for providing insight into work to date, as well as pointing towards the key aspects that merit further investigation.

Acknowledgments

This research was supported in part by the Australian Research Council, the Kone Foundation and the Academy of Finland. We would like to thank Kimmo Koskenniemi for many valuable discussions and comments concerning the early phases of the features and the methods sections.

References

Abainia, K., Ouamour, S., & Sayoud, H. (2014). Robust Language Identification of Noisy Texts - Proposal of Hybrid Approaches. In *25th International Workshop on Database and Expert Systems Applications (DEXA)*, pp. 228–232, Munich, Germany.

Abainia, K., Ouamour, S., & Sayoud, H. (2016). Effective Language Identification of Forum Texts Based on Statistical Approaches. *Information Processing and Management*, *52*, 491–512.

Abney, S., & Bird, S. (2010). The Human Language Project: Building a Universal Corpus of the World's Languages. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pp. 88–97, Los Angeles, USA.

Ács, J., Grad-Gyenge, L., Bruno, T., & Oliveira, R. d. R. (2015). A Two-level Classifier for Discriminating similar Languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 73–77, Hissar, Bulgaria.

Adams, G., & Resnik, P. (1997). A Language Identification Application Built on the Java Client/server Platform. In *Proceedings of the ACL/EACL'97 Workshop on From Research to Commercial Applications: Making NLP Work in Practice*, pp. 43–47, Madrid, Spain.

Adouane, W. (2016). Automatic Detection of Underresourced Languages: Dialectal Arabic Short Texts. Master's thesis, University of Gothenburg, Gothenburg, Sweden.

Adouane, W., & Dobnik, S. (2017). Identification of Languages in Algerian Arabic Multilingual Documents. In *Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP 2017)*, pp. 1–8, Valencia, Spain.

Adouane, W., Semmar, N., & Johansson, R. (2016a). Arabicized and Romanized Berber Automatic Identification. In *Proceedings of the International Conference on Information and Communication Technologies for Amazingh (TICAM 2016)*, Rabat, Morocco. IRCAM.

Adouane, W., Semmar, N., & Johansson, R. (2016b). ASIREM Participation at the Discriminating Similar Languages Shared Task 2016. In *Proceedings of the Third Workshop*

on NLP for Similar Languages, Varieties and Dialects (VarDial), pp. 163–169, Osaka, Japan.

Adouane, W., Semmar, N., & Johansson, R. (2016c). Romanized Berber and Romanized Arabic Automatic Language Identification Using Machine Learning. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 53–61, Osaka, Japan.

Adouane, W., Semmar, N., Johansson, R., & Bobicev, V. (2016d). Automatic Detection of Arabicized Berber and Arabic Varieties. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 63–72, Osaka, Japan.

Ahmed, B., Cha, S.-H., & Tappert, C. (2004). Language Identification from Text Using N-gram Based Cumulative Frequency Addition. In *Proceedings of Student/Faculty Research Day*, pp. 12.1–12.8, CSIS, Pace University, New York, USA.

Akhil, M. B. S. S., & Abhishek, J. (2014). Language Identification, Transliteration and Resolving Common Words Ambiguity in a Pair of Languages: Shared Task on Transliterated Search. In *Working Notes of Shared Task on Transliterated Search at Forum for Information Retrieval Evaluation (FIRE'14)*, Bangalore, India.

Akhtyamova, L., Cardiff, J., & Ignatov, A. (2017). Twitter Author Profiling Using Word Embeddings and Logistic Regression - Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Akosu, N., & Selamat, A. (2014). A Dynamic Model Selection Algorithm for Language Identification of Under-resourced Languages. *International Journal of Digital Content Technology and its Applications (JDCTA)*, *8*.

Al-Badrashiny, M., & Diab, M. T. (2016). LILI: A Simple Language Independent Approach for Language Identification. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016): Technical Papers*, pp. 1211–1219, Osaka, Japan.

Al-Badrashiny, M., Elfardy, H., & Diab, M. (2015). AIDA2: A Hybrid Approach for Token and Sentence Level Dialect Identification in Arabic. In *Proceedings of the 19th Conference on Computational Language Learning*, pp. 42–51, Beijing, China.

Alex, B. (2005). An Unsupervised System for Identifying English Inclusions in German Text. In *Proceedings of the Student Research Workshop, ACL-05*, pp. 133–138, Ann Arbor, Michigan, USA.

Alex, B., Dubey, A., & Keller, F. (2007). Using Foreign Inclusion Detection to Improve Parsing Performance. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning 2007 (EMNLP-CoNLL 2007)*, pp. 151–160, Prague, Czech Republic.

Alfter, D. (2015). Language Segmentation. Master's thesis, Universität Trier, Trier, Germany.

Ali, A., Dehak, N., Cardinal, P., Khurana, S., Yella, S. H., Glass, J., Bell, P., & Renals, S. (2016). Automatic Dialect Detection in Arabic Broadcast Speech. In *Proceedings of Interspeech 2016*, pp. 2934–2938, San Francisco, USA.

Ali, A., Vogel, S., & Renals, S. (2017). Speech Recognition challenge in the Wild: Arabic MGB-3. *arXiv preprint*, *arXiv:1709.07276*.

Alrifai, K., Rebdawi, G., & Ghneim, N. (2017). Arabic Tweeps Gender and Dialect Prediction – Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Alshutayri, A., & Atwell, E. (2017). Exploring Twitter as a Source of an Arabic Dialect Corpus. *International Journal of Computational Linguistics (IJCL)*, *8*(2), 37–44.

Alshutayri, A., Atwell, E., Alosaimy, A., Dickins, J., Ingleby, M., & Watson, J. (2016). Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 204–211, Osaka, Japan.

Amine, A., Elberrichi, Z., & Simonet, M. (2010). Automatic Language Identification: An Alternative Unsupervised Approach Using a New Hybrid Algorithm. *International Journal of Computer Science and Applications*, *7*(1), 94–107.

Anand, S. (2014). Language Identification for Transliterated Forms of Indian Language Queries. In *Working Notes of Forum for Information Retrieval Evaluation (FIRE)*, Bangalore, India.

Anstein, S. (2013). *Computational Approaches to the Comparison of Regional Variety Corpora: Prototyping a Semi-automatic System for German*. Ph.D. thesis, University of Stuttgart.

Armstrong-Warwick, S., Thompson, S., McKelvie, D., & Petitpierre, D. (1994). Data in your language: the ECI Multilingual Corpus 1. In *Proceedings of International Workshop on Sharable Natural Language Resources*, pp. 97–106, Ikoma, Japan.

Artemenko, O., & Shramko, M. (2005). Entwicklung eines Werkzeugs zur Sprachidentifikation in mono- und multilingualen Texten. Master's thesis, Universität Hildesheim.

Babu, A. S., & Kumar, P. (2010). Comparing Neural Network Approach with N-Gram Approach for Text Categorization. *International Journal on Computer Science and Engineering*, *2*(1), 80–83.

Balažević, I., Braun, M., & Müller, K.-R. (2016). Language Detection For Short Text Messages In Social Media. *arXiv preprint*, *arXiv:1608.08515*.

Baldwin, T., & Lui, M. (2010a). Language Identification: The Long and the Short of the Matter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pp. 229–237, Los Angeles, USA.

Baldwin, T., & Lui, M. (2010b). Multilingual Language Identification: ALTW 2010 Shared Task Dataset. In *Proceedings of the Australasian Language Technology Workshop 2010 (ALTW 2010)*, pp. 5–7, Melbourne, Australia.

Banerjee, S., Roy, A., Kuila, A., Naskar, S. K., Bandyopadhyay, S., & Rosso, P. (2014). A Hybrid Approach for Transliterated Word-Level Language Identification: CRF with Post Processing Heuristics. In *Proceedings of the Sixth Workshop of the Forum for Information Retrieval Evaluation (FIRE 2014)*, pp. 54–59, Bangalore, India.

Bar, K., & Dershowitz, N. (2014). The Tel Aviv University System for the Code-Switching Workshop Shared Task. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 139–143, Doha, Qatar.

Barbaresi, A. (2016). An Unsupervised Morphological Criterion for Discriminating Similar Languages. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 212–220, Osaka, Japan.

Barbaresi, A. (2017). Discriminating between Similar Languages using Weighted Subword Features. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects*, pp. 184–189, Valencia, Spain.

Barkov, A. (2008). *mguesser*. Software available at `https://github.com/yaoweibin/mguesser` (not updated since 2011).

Barman, U., Das, A., Wagner, J., & Foster, J. (2014a). Code Mixing: A Challenge for Language Identification in the Language of Social Media. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 127–132, Doha, Qatar.

Barman, U., Wagner, J., Chrupala, G., & Foster, J. (2014b). DCU-UVT: Word-Level Language Classification with Code-Mixed Data. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 13–23, Doha, Qatar.

Basile, A., Dwyer, G., Medvedeva, M., Rawee, J., Haagsma, H., & Nissim, M. (2017). N-GrAM: New Groningen Author-profiling Model—Notebook for PAN at CLEF 2017. In *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland.

Batchelder, E. O. (1992). A Learning Experience: Training an Artificial Neural Network to Discriminate Languages. Technical report.

Baykan, E., Henzinger, M., & Weber, I. (2008). Web page language identification based on URLs. In *Proceedings of the 34th International Conference on Very Large Data Bases (VLDB 2008)*, pp. 176–187, Auckland, New Zealand.

Beesley, K. R. (1988). Language Identifier: A Computer Program for Automatic Natural-Language Identification of On-line Text. In *Proceedings of the 29th Annual Conference of the American Translators Association: Languages at Crossroads*, pp. 47–54, Seattle, USA.

Bekavac, B., Kocijan, K., & Tadić, M. (2014). Near Language Identification using NooJ. In *Formalising Natural Languages with NooJ 2014: Selected papers from the NooJ 2014 International Conference*, pp. 152–166. Cambridge Scholars Publishing, Sassari, Italy.

Benedetto, D., Caglioti, E., & Loreto, V. (2002). Language Trees and Zipping. *Physical Review Letters*, *88*(4).

Bergsma, S., McNamee, P., Bagdouri, M., Fink, C., & Wilson, T. (2012). Language Identification for Creating Language-specific Twitter Collections. In *Proceedings of the*

Second Workshop on Language in Social Media (LSM2012), pp. 65–74, Montréal, Canada.

Bestgen, Y. (2017). Improving the Character Ngram Model for the DSL Task with BM25 Weighting and Less Frequently Used Feature Sets. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 115–123, Valencia, Spain.

Bezdek, J. C., Ehrlich, R., & Full, W. (1984). FCM: The Fuzzy c-Means Clustering Algorithm. *Computers and Geosciences*, *10*(2-3), 191–203.

Bhargava, A., & Kondrak, G. (2010). Language Identification of Names with SVMs. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pp. 693–696, Los Angeles, California, USA.

Bhargava, R., Sharma, Y., Sharma, S., & Baid, A. (2015). Query Labeling for Indic Languages Using a Hybrid Approach. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015)*, pp. 42–44, Gandhinagar, India.

Bhattu, S. N., & Ravi, V. (2015). Language Identification in Mixed Script Social Media Text. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015)*, pp. 39–31, Gandhinagar, India.

Biemann, C., & Teresniak, S. (2005). Disentangling from Babylonian confusion –- Unsupervised Language Identification. In Gelbukh, A. (Ed.), *Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2005)*, pp. 773–784, Mexico City, Mexico. Springer.

Bilcu, E. B., & Astola, J. (2006). A Hybrid Neural Network for Language Identification from Text. In *Proceedings of the 16th IEEE Signal Processing Society Workshop on Machine Learning for Signal Processing*, pp. 253–258, Maynooth, Ireland.

Binas, A. (2005). Markovian Time Series Models for Language Identification. Project Report.

Bisani, M., & Ney, H. (2008). Joint-sequence Models for Grapheme-to-phoneme Conversion. *Speech Communication*, *50*(5), 434–451.

Bjerva, J. (2016). Byte-based Language Identification with Deep Convolutional Networks. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 119–126, Osaka, Japan.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Blodgett, S. L., Wei, J. T.-Z., & O'Connor, B. (2017). A Dataset and Classifier for Recognizing Social Media English. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 56–61, Copenhagen, Denmark.

Bobicev, V. (2015). Discriminating between Similar Languages Using PPM. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 59–65, Hissar, Bulgaria.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory COLT' 92*, pp. 144–152, Pittsburgh, USA.

Botha, G., Zimu, V., & Barnard, E. (2007). Text-based Language Identification for South African Languages. *Transactions of the South African Institute of Electrical Engineers*, *98*(4), 141–148.

Botha, G. R. (2008). Text-Based Language Identification for The South African Languages. Master's thesis, University of Pretoria, Hatfield, Pretoria, South Africa.

Botha, G. R., & Barnard, E. (2007). Factors that Affect the Accuracy of Text-based Language Identification. In Tapamo, J. R., & Nicolls, F. (Eds.), *Proceedings of the Eighteenth Annual Symposium of the Pattern Recognition Association of South Africa*, pp. 7–12, Pietermaritzburg, South Africa.

Botha, G. R., & Barnard, E. (2012). Factors that Affect the Accuracy of Text-based Language Identification. *Computer Speech and Language*, *26*(5), 307–320.

Bouamor, H., Habash, N., & Oflazer, K. (2014). A Multidialectal Parallel Corpus of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pp. 1240–1245, Reykjavik, Iceland.

Bošnjak, M., Rodrigues, E. M., & Sarmento, L. (2013). Robust Language Identification with RapidMiner: A Text Mining Use Case. In Hofmann, M., & Klinkenberg, R. (Eds.), *RapidMiner: Data Mining Use Cases and Business Analytics Applications*, pp. 213–239. Chapman and Hall/CRC.

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, *24*(1), 123–140.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

Brown, P. F., deSouza, P. V., Mercer, R. L., Della Pietra, V. J., & Lai, J. C. (1992). Class-Based n-gram Models of Natural Language. *Computational Linguistics*, *18*(4), 467–479.

Brown, R. D. (2012). Finding and Identifying Text in 900+ Languages. *Digital Investigation*, *9*, S34–S43.

Brown, R. D. (2013). Selecting and Weighting N-grams to Identify 1100 Languages. In *Proceedings of the 16th International Conference on Text, Speech and Dialogue (TSD 2013)*, pp. 475–483, Plzeň, Czech Republic.

Brown, R. D. (2014a). *Language-Aware String Extractor*. Software available at `https://sourceforge.net/projects/la-strings/` (last updated on February 2018).

Brown, R. D. (2014b). Non-linear Mapping for Improved Identification of 1300+ Languages. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pp. 627–632, Doha, Qatar.

Bush, B. O. (2014). Language Identification of Tweets Using LZW Compression. In *3rd Pacific Northwest Regional NLP Workshop (NW-NLP 2014)*, Redmond, USA.

Cann, P. v. (2015). Dialect Identification on Twitter: A Research About the Detection of the Limburgian Dialect from Twitter messages. Master's thesis, University of Tilburg.

Carter, S., Weerkamp, W., & Tsagkias, M. (2013). Microblog Language Identification: Overcoming the Limitations of Short, Unedited and Idiomatic text. *Language Resources and Evaluation*, *47*(1), 195–215.

Castro, D., Souza, E., & de Oliveira, A. L. I. (2016). Discriminating between Brazilian and European Portuguese National Varieties on Twitter Texts. In *Proceedings of the 5th Brazilian Conference on Intelligent Systems (BRACIS)*, pp. 265–270, Recife, Pernambuco, Brazil. IEEE.

Castro, D. W., Souza, E., Vitório, D., Santos, D., & Oliveira, A. L. I. (2017). Smoothed N-gram Based Models for Tweet Language Identification: A Case Study of the Brazilian and European Portuguese National Varieties. *Applied Soft Computing*, *61*, 1160–1172.

Cavnar, W. B., & Trenkle, J. M. (1994). N-Gram-Based Text Categorization. In *Proceedings of SDAIR-94, Third Annual Symposium on Document Analysis and Information Retrieval*, pp. 161–175, Las Vegas, USA.

Cazamias, J., Dixit, C., & Marek, M. (2015). Large-Scale Language Classification - Writing a Detector for 200 Languages on Twitter. Stanford course report.

Çöltekin, C., & Rama, T. (2016). Discriminating Similar Languages: Experiments with Linear SVMs and Neural Networks. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 15–24, Osaka, Japan.

Çöltekin, C., & Rama, T. (2017). Tübingen System in VarDial 2017 Shared Task: Experiments with Language Identification and Cross-lingual Parsing. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 146–155, Valencia, Spain.

Celikel, E. (2005). Language Discrimination via PPM Model. In Selvaraj, H., & Srimani, P. K. (Eds.), *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*, Vol. 1, pp. 57–62, Las Vegas, Nevada, USA.

Ceylan, H., & Kim, Y. (2009). Language Identification of Search Engine Queries. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 1066–1074, Suntec, Singapore.

Chanda, A., Das, D., & Mazumdar, C. (2016a). Columbia-Jadavpur submission for EMNLP 2016 Code-Switching Workshop Shared Task: System Description. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 112–115, Austin, TX, USA.

Chanda, A., Das, D., & Mazumdar, C. (2016b). Unraveling the English-Bengali Code-Mixing Phenomenon. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 80–89, Austin, TX, USA.

Chang, J. C., & Lin, C.-C. (2014). Recurrent-neural-network for Language Detection on Twitter Code-Switching Corpus. *arXiv preprint*, *arXiv:1412.4314*.

Chen, S. F., & Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, *13*(4), 359–394.

Chen, S. F., & Maison, B. (2003). Using Place Name Data to Train Language Identification Models. In *8th European Conference on Speech Communication and Technology EUROSPEECH 2003 - INTERSPEECH 2003*, pp. 1349–1352, Geneva, Switzerland.

Chen, T., & Guestrin, C. (2016). XGBoost: Reliable Large-sclae Tree Boosting System. *arXiv preprint*, *1603.02754*, 1–6.

Chen, Y., You, J., Chu, M., Zhao, Y., & Wang, J. (2006). Identifying Language Origin of Person Names with N-grams of Different Units. In *Proceedings of the 2006 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2006)*, Vol. 1, pp. 729–732, Toulouse, France.

Chew, Y. C., Mikami, Y., Marasinghe, C. A., & Nandasara, S. T. (2009). Optimizing n-gram Order of an n-gram Based Language Identification Algorithm for 68 Written Languages. *International Journal on Advances in ICT for Emerging Regions (ICTer)*, *02*(02), 21–28.

Chew, Y. C., Mikami, Y., & Nagano, R. L. (2011). Language Identification of Web Pages Based on Improved N-gram Algorithm. *International Journal of Computer Science Issues*, *8*(3), 47–58.

Chittaranjan, G., Vyas, Y., Bali, K., & Choudhury, M. (2014). Word-level Language Identification using CRF: Code-switching Shared Task Report of MSR India System. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 73–79, Doha, Qatar.

Church, K. (1985). Stress Assignment in Letter to Sound Rules for Speech Synthesis. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pp. 246–253, Chicago, Illinois, USA.

Cianflone, A., & Kosseim, L. (2016). N-gram and Neural Language Models for Discriminating Similar Languages. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 243–250, Osaka, Japan.

Ciobanu, A. M., & Dinu, L. P. (2016). A Computational Perspective on the Romanian Dialects. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pp. 3281–3285, Portorož, Slovenia.

Ciobanu, A. M., Zampieri, M., Malmasi, S., & Dinu, L. P. (2017). Including Dialects and Language Varieties in Author Profiling. *arXiv preprint*, *arXiv:1707.00621*.

Clematide, S., & Makarov, P. (2017). CLUZH at VarDial GDI 2017: Testing a Variety of Machine Learning Tools for the Classification of Swiss German Dialects. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 170–177, Valencia, Spain.

Clyne, M. (1992). *Pluricentric Languages: Different Norms in Different Nations*. CRC Press, Boca Raton, USA.

Cohen, W. W. (1995). Fast Effective Rule Induction. In *Proceedings of the Twelfth International Conference on Machine Learning (ML95)*, pp. 115–123, Tahoe City, California, USA.

Constable, P., & Simons, G. (2000). Language identification and IT: Addressing Problems of Linguistic Diversity on a Global Scale. Sil electronic working papers 2000-001, SIL International, Dallas, USA.

Cotterell, R., & Callison-Burch, C. (2014). A Multi-Dialect, Multi-Genre Corpus of Informal Written Arabic. In *Proceedings of the 9th Language Resources and Evaluation Conference (LREC 2014)*, Reykjavik, Iceland.

Cowie, J., Ludovik, Y., & Zacharski, R. (1999). Language Recognition for Mono- and Multilingual Documents. In *Proceedings of the VexTal Conference*, pp. 209–214, Venice, Italy.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, *7*, 551–585.

Creutz, M. (2003). Unsupervised Segmentation of Words Using Prior Distributions of Morph Length and Frequency. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, Vol. 1, pp. 280–287, Sapporo, Japan.

Criscuolo, M., & Aluísio, S. M. (2017). Discriminating between Similar Languages with Word-level Convolutional Neural Networks. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 124–130, Valencia, Spain.

da Silva, J. F., & Lopes, G. P. (2006a). Identification of Document Language in Hard Contexts. In *ACM-SIGIR 2006 New Directions in Multilingual Information Access Proceedings of the Workshop*, pp. 40–48, Seattle, USA.

da Silva, J. F., & Lopes, G. P. (2006b). Identification of Document Language is Not yet a Completely Solved Problem. In *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, p. 212, Sydney, Australia. IEEE.

da Silva, J. F., & Lopes, G. P. (2007). Using Covariance as a Similarity Measure for Document Language Identification in Hard Contexts. *Pliska Studia Mathematica Bulgarica*, *18*, 341–360.

Darwish, K., Sajjad, H., & Mubarak, H. (2014). Verifiably Effective Arabic Dialect Identification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pp. 1465–1468, Doha, Qatar.

Das, A., & Gambäck, B. (2013). Code-Mixing in Social Media Text: The Last Language Identification Frontier?. *Traitement Automatique des Langues*, *54*(3), 41–64.

Das, A., & Gambäck, B. (2014). Identifying Languages at the Word Level in Code-Mixed Indian Social Media Text. In *Proceedings of the 11th International Conference on Natural Language Processing*, pp. 169–178, Goa, India.

Debole, F., & Sebastiani, F. (2005). An Analysis of the Relative Hardness of Reuters-21578 Subsets. *Journal of the American Society for Information Science and technology*, *56*(6), 584–596.

Diaconis, P., & Graham, R. L. (1977). Spearmam's Footrule as a Measure of Disarray. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*(2), 262–268.

Diwersy, S., Evert, S., & Neumann, S. (2014). A Weakly Supervised Multivariate Approach to the Study of Language Variation. In Szmrecsanyi, B., & Wälchli, B. (Eds.), *Aggregating Dialectology, Typology, and Register Analysis. Linguistic Variation in Text and Speech*. De Gruyter, Berlin.

Dongen, N. (2017). Analysis and Prediction of Dutch-English Code-switching in Dutch Social Media Messages. Master's thesis, Universiteit van Amsterdam, Amsterdam, Netherlands.

Doval, Y., Vilares, D., & Vilares, J. (2014). Automatic Language Identification in Twitter: Adapting State-of-the-Art Identifiers to the Iberian Context. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, pp. 39–43, Girona, Spain.

Doyle, G. (2014). Mapping Dialectal Variation by Querying Social Media. In *Proceedings of the 14th Conference of the EACL (EACL 2014)*, pp. 98–106, Gothenburg, Sweden.

Dunning, T. (1994). Statistical Identification of Language. Tech. rep. MCCS 940-273, Computing Research Laboratory, New Mexico State University.

Dutta, S., Saha, T., Banerjee, S., & Naskar, S. K. (2015). Text Normalization in Code-Mixed Social Media Text. In *2nd International Conference on Recent Trends in Information Systems (ReTIS)*, pp. 378–382, Kolkata, India.

Duvenhage, B., Ntini, M., & Ramonyai, P. (2017). Improved Text Language Identification for the South African Languages. In *Proceedings of the 28th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2017)*, Bloemfontein, South Africa.

El-Shishiny, H., Troussov, A., McCloskey, D. J., Takeuchi, M., Nevidomsky, A., & Volkov, P. (2004). Word Fragments Based Arabic Language Identification. In *Proceedings from NEMLAR International Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

Eldesouki, M., Dalvi, F., Sajjad, H., & Darwish, K. (2016). QCRI DSL 2016: Spoken Arabic Dialect Identification Using Textual Features. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 221–226, Osaka, Japan.

Elfardy, H., Al-Badrashiny, M., & Diab, M. (2013). Code Switch Point Detection in Arabic. In Métais, E., Meziane, F., Sararee, M., Sugumaran, V., & Vadera, S. (Eds.), *Proceedings of the Natural Language Processing and Information Systems - 18th International Conference on Applications of Natural Language to Information Systems (NLDB 2013)*, pp. 412–416, Salford, UK. Springer.

Elfardy, H., Al-Badrashiny, M., & Diab, M. (2014). AIDA: Identifying Code Switching in Informal Arabic Text. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 94–101, Doha, Qatar.

Elfardy, H., & Diab, M. (2012). Token Level Identification of Linguistic Code Switching. In Kay, M., & Boitet, C. (Eds.), *Proceedings of COLING 2012: Posters*, pp. 287–296, Mumbai, India. The COLING 2012 Organizing Committee.

Elfardy, H., & Diab, M. (2013). Sentence Level Dialect Identification in Arabic. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 456–461, Sofia, Bulgaria.

Elgabou, H. A., & Kazakov, D. (2017). Building Dialectal Arabic Corpora. In *The Proceedings of the First Workshop on Human-Informed Translation and Interpreting Technology (HiT-IT)*, pp. 52–57, Varna, Bulgaria.

Elworthy, D. (1998). Language Identification with Confidence Limits. In *Proceedings of the 6th Annual Workshop on Very Large Corpora*, pp. 94–101, Montréal, Canada.

Emerson, G., Tan, L., Fertmann, S., Palmer, A., & Regneri, M. (2014). SeedLing: Building and Using a Seed corpus for the Human Language Project. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pp. 77–85, Baltimore, USA.

Eskander, R., Al-Badrashiny, M., Habash, N., & Rambow, O. (2014). Foreign Words and the Automatic Processing of Arabic Social Media Text Written in Roman Script. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 1–12, Doha, Qatar.

Espichán-Linares, A., & Oncevay-Marcos, A. (2017). A Low-Resourced Peruvian Language Identification Model. In Lossio-Ventura, J. A., & Alatrista-Salas, H. (Eds.), *Proceedings of the 4th Annual International Symposium on Information Management and Big Data (SIMBig 2017)*, pp. 57–63, Lima, Peru.

Espichán-Linares, A., & Oncevay-Marcos, A. (2018). Language Identification with Scarce Data: A Case Study from Peru. In *Proceedings of the 4th Annual International Symposium on Information Management and Big Data (SIMBig 2017), Revised Selected Papers*, Lima, Peru.

Fabra-Boluda, R., Rangel, F., & Rosso, P. (2015). NLEL UPV Autoritas participation at Discrimination between Similar Languages DSL 2015 Shared Task. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 52–58, Hissar, Bulgaria.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., & Lin, C.-J. (2008). LIBLINEAR: A Library for Large Linear Classification. *Journal of machine learning research*, *9*(Aug), 1871–1874.

Franco-Penya, H.-H., & Sanchez, L. M. (2016). Tuning Bayes Baseline for Dialect Detection. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 227–234, Osaka, Japan.

Franco-Salvador, M., Kondrak, G., & Rosso, P. (2017a). Bridging the Native Language and Language Variety Identification Tasks. In *Proceedings of the 21st International Conference Knowledge-Based and Intelligent Information and Engineering Systems (KES-2017)*, Vol. 112, pp. 1554–1561, Marseille, France.

Franco-Salvador, M., Plotnikova, N., Pawar, N., & Benajiba, Y. (2017b). Subword-based Deep Averaging Networks for Author Profiling – Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Franco-Salvador, M., Rangel, F., Rosso, P., Taulé, M., & Martít, M. A. (2015a). Language Variety Identification Using Distributed Representations of Words and Documents. In *Proceedings of the 6th International Conference of the CLEF Association (CLEF' 15): Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pp. 28–40.

Franco-Salvador, M., Rosso, P., & Rangel, F. (2015b). Distributed Representations of Words and Documents for Discriminating Similar Languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 11–16, Hissar, Bulgaria.

Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, *55*, 119–139.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, *28*(2), 337–407.

Gamallo, P., Garcia, M., & Sotelo, S. (2014). Comparing Ranking-based and Naive Bayes Approaches to Language Detection on Tweets. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, pp. 12–16, Girona, Spain.

Gamallo, P., Pichel, J. R., & Alegria, I. (2017). A Perplexity-Based Method for Similar Languages Discrimination. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 109–114, Valencia, Spain.

Gamallo, P., Pichel, J. R., Alegria, I., & Agirrezabal, M. (2016). Comparing two Basic Methods for Discriminating Between Similar Languages and Varieties. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 170–177, Osaka, Japan.

Garg, A., Gupta, V., & Jindal, M. (2014). A Survey of Language Identification Techniques and Applications. *Journal of Emerging Technologies in Web Intelligence*, *6*(4), 388–400.

Ghosh, S., Ghosh, S., & Das, D. (2015). Labeling of Query Words using Conditional Random Field. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015)*, pp. 31–34, Gandhinagar, India.

Giguet, E. (1995). Categorization according to Language: A step toward combining Linguistic Knowledge and Statistic Learning. In *Proceedings of the International Workshop on Parsing Technologies (IWPT'95)*, Prague - Karlovy Vary, Czech Republic.

Giguet, E. (1998). *Méthode pour l' Analyse Automatique de Structures Formelles sur Documents Multilingues*. Ph.D. thesis, Université de Caen.

Giwa, O. (2016). *Language Identification for Proper Name Pronunciation*. Ph.D. thesis, North-West University, Vaal Triangle.

Giwa, O., & Davel, M. H. (2013). N-Gram based Language Identification of Individual Words. In Robinson, P. (Ed.), *Proceedings of the 24th Annual Symposium of the Pattern Recognition Association of South Africa*, pp. 15–22, Johannesburg, South Africa.

Giwa, O., & Davel, M. H. (2014). Language Identification of Individual Words with Joint Sequence Models. In *Proceedings of Interspeech 2014*, Singapore.

Gold, E. M. (1967). Language Identification in the Limit. *Information and Control*, *10*(5), 447–474.

Goldszmidt, M., Najork, M., & Paparizos, S. (2013). Boot-Strapping Language Identifiers for Short Colloquial Postings. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2013), Part II*, pp. 95–111, Prague, Czech Republic. Springer.

Google (2019). Google Chrome web browser. `http://www.google.com/chrome`. Accessed: 2019-07-17.

Gottron, T., & Lipka, N. (2010). A Comparison of Language Identification Approaches on Short, Query-style Texts. In *Advances in Information Retrieval - Proceedings of the 32nd annual European Conference on Information Retrieval Research (ECIR 2010)*, pp. 611–614, Milton Keynes, UK. Springer.

Goutte, C., Léger, S., Malmasi, S., & Zampieri, M. (2016). Discriminating Similar Languages: Evaluations and Explorations. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*.

Goutte, C., & Léger, S. (2015). Experiments in Discriminating Similar Languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 78–84, Hissar, Bulgaria.

Goutte, C., & Léger, S. (2016). Advances in Ngram-based Discrimination of Similar Languages. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 178–184, Osaka, Japan.

Goutte, C., Léger, S., & Carpuat, M. (2014). The NRC System for Discriminating Similar Languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, pp. 139–145, Dublin, Ireland.

Grefenstette, G. (1995). Comparing Two Language Identification Schemes. In *Proceedings of the 3rd International conference on Statistical Analysis of Textual Data (JADT 1995)*, Rome, Italy.

Grouin, C., Forest, D., Da Sylva, L., Paroubek, P., & Zweigenbaum, P. (2011). Présentation et Résultats du Défi Fouille de Texte DEFT2010 Où et Quand un Article de Presse a-t-il Été Écrit?. In *Actes du sixième Défi Fouille de Textes*, pp. 3–14, Montpellier, France.

Guellil, I., & Azouaou, F. (2016). Arabic Dialect Identification With an Unsupervised Learning (based on a lexicon). Application case: Algerian Dialect. In *Proceedings of the 2016 IEEE International Conference on Computational Science and Engineering and IEEE International Conference on Embedded and Ubiquitous Computing and 15th*

Intl Symposium on Distributed Computing and Applications for Business Engineering (CSE-EUC-DCABES 2016), pp. 724–731, Paris, France.

Gupta, B., Bhatt, G., & Mittal, A. (2016). Language Identification and Disambiguation in Indian Mixed-Script. In Bjørner, N., Prasad, S., & Parida, L. (Eds.), *Distributed Computing and Internet Technology*, pp. 113–121. Springer.

Gupta, D. K., Kumar, S., & Ekbal, A. (2014). Machine Learning Approach for Language Identification & Transliteration: Shared Task Report of IITP-TS. In *Forum for Information Retrieval Evaluation (FIRE)*, pp. 60–64, Bangalore, India.

Guzmán, G. A., Ricard, J., Serigos, J., Bullock, B., & Toribio, A. J. (2017). Moving Code-switching Research Toward More Empirically Grounded Methods. In Declerck, T., & Kübler, S. (Eds.), *Proceedings of the Workshop on Corpora in the Digital Humanities (CDH 2017)*, pp. 1–9, Bloomington, IN, USA.

Gómez-Adorno, H., Markov, I., Baptista, J., Sidorov, G., & Pinto, D. (2017). Discriminating between Similar Languages Using a Combination of Typed and Untyped Character N-grams and Words. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 137–145, Valencia, Spain.

Habash, N. (2007). Arabic Morphological Representations for Machine Translation. *Text, Speech and Language Technology*, *38*, 263–285.

Habash, N., & Sadat, F. (2006). Arabic Preprocessing Schemes for Statistical Machine Translation. In *Proceedings of the 7th Meeting of the North American Chapter of the Association for Computational Linguistics/Human Language Technologies Conference (HLT-NAACL06)*, pp. 49–52, New York, NY, USA.

Haţegan, A., Bârligă, B., & Tăbuş, I. (2009). Language Identification of Individual Words in a Multilingual Automatic Speech Recognition System. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, pp. 4357–4360, Taipei, Taiwan. IEEE.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *ACM SIGKDD explorations newsletter*, *11*(1), 10–18.

Hammarström, H. (2007). A Fine-Grained Model for Language Identification. In *Proceedings of Improving Non English Web Searching (iNEWS-07) Workshop at SIGIR 2007*, pp. 14–20, Amsterdam, Netherlands.

Hammerl, T. (2013). *JTCL*. Software available at `http://textcat.sourceforge.net/` (not updated since first release).

Hamzah, A. (2010). Deteksi bahasa untuk dokumen teks berbahasa Indonesia. In *Seminar Nasional Informatika 2010 (semnasIF 2010)*, pp. A5–A13, Jakarta, Indonesia.

Hanani, A., Qaroush, A., & Taylor, S. (2016). Classifying ASR Transcriptions According to Arabic Dialect. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 126–134, Osaka, Japan.

Hanani, A., Qaroush, A., & Taylor, S. (2017). Identifying Dialects with Textual and Acoustic Cues. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 93–101, Valencia, Spain.

Hanif, F., Latif, F., & Khiyal, M. S. H. (2007). Unicode Aided Language Identification across Multiple Scripts and Heterogeneous Data. *Information Technology Journal*, *6*(4), 534–540.

Hasimu, M., & Silamu, W. (2017). Three-stage Short Text Language Identification Algorithm. *Journal of Digital Information Management*, *15*(6), 354–371.

Hayati, K. (2004). Language Identification on the World Wide Web. Master's thesis, University of California Santa Cruz, Santa Cruz, California, USA.

Hayta, S. B., Takçı, H., & Eminli, M. (2013). Language Identification Based On N-Gram Feature Extraction Method By Using Classifiers. *IU-Journal of Electrical and Electronics Engineering*, *13*(2), 1629–1639.

He, J., Zhang, Z., Zhao, X., Li, P., & Yan, Y. (2016). Similar Language Identification for Uyghur and Kazakh on Short Spoken Texts. In *Proceedings of the 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC 2016)*, Vol. 2, pp. 496–499, Hangzhou, China.

Hecht-Nielsen, R. (1989). Theory of the Backpropagation Neural Network. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 1989)*, pp. I593–I605, Washington, DC, USA.

Henrich, P. (1989). Language Identification for the Automatic Grapheme-to-phoneme Conversion of Foreign Words in a German Text-to-speech System. In *First European Conference on Speech Communication and Technology*, pp. 2220–2223, Paris, France.

Herman, O., Suchomel, V., Baisa, V., & Rychlý, P. (2016). DSL Shared task 2016: Perfect Is The Enemy of Good Language Discrimination Through Expectation-Maximization and Chunk-based Language Model. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 114–118, Osaka, Japan.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*, 1735–1780.

Hollenstein, N., & Aepli, N. (2015). A Resource for Natural Language Processing of Swiss German Dialects. In *Proceedings of GSCL*, pp. 108–109.

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression* (3rd ed edition). Wiley Series in Probability and Statistics. Wiley, Hoboken, N.J., USA.

House, A. S., & Neuburg, E. P. (1977). Toward Automatic Identification of the Language of an Utterance. I. Preliminary Methodological Considerations. *The Journal of the Acoustical Society of America*, *62*(3), 708–713.

Huang, C.-R., & Lee, L.-H. (2008). Contrastive Approach towards Text Source Classification based on Top-Bag-of-Word Similarity. In *Proceedings of the 22nd Pacific Asia Conference on Language, Information and Computation*, pp. 404–410, Cebu City, Philippines.

Huang, F. (2015). Improved Arabic Dialect Classification with Social Media Data. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pp. 2118–2126, Lisbon, Portugal.

Hughes, B., Baldwin, T., Bird, S., Nicholson, J., & MacKinlay, A. (2006). Reconsidering Language Identification for Written Language Resources. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pp. 485–488, Genoa, Italy.

Hurtado, L.-F., Pla, F., Giménez, M., & Sanchis, E. (2014). ELiRF-UPV at TweetLID: Twitter Language Identification. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, pp. 35–38, Girona, Spain.

Häkkinen, J., & Tian, J. (2001). N-gram and Decision Tree Based Language Identification for Written Words. In *Conference Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 2001)*, pp. 335–338, Madonna di Campiglio, Italy.

Indhuja, K., Indu, M., Sreejith, C., & Reghu Raj, P. C. (2014). Text Based Language Identification System for Indian Languages Following Devanagiri Script. *International Journal of Engineering Reseach and Technology, 3(4)*, 327–331.

Ionescu, R. T. (2013). Local rank distance. In Björner, N., Negru, V., Ida, T., Jebelean, T., Petcu, D., Watt, S., & Zaharie, D. (Eds.), *Proceedings of the 15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2013)*, pp. 219–226, Timisoara, Romania.

Ionescu, R. T., & Butnaru, A. M. (2017). Learning to Identify Arabic and German Dialects using Multiple Kernels. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 200–209, Valencia, Spain.

Ionescu, R. T., & Popescu, M. (2016). UnibucKernel: An Approach for Arabic Dialect Identification based on Multiple String Kernels. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 135–144, Osaka, Japan.

Jaech, A., Mulcaire, G., Hathi, S., Ostendorf, M., & Smith, N. A. (2016a). A Neural Model for Language Identification in Code-Switched Tweets. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 60–64, Austin, TX, USA.

Jaech, A., Mulcaire, G., Hathi, S., Ostendorf, M., & Smith, N. A. (2016b). Hierarchical Character-Word Models for Language Identification. In *Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media*, pp. 84–93, Austin, TX, USA.

Jain, D. (2015). DA-IICT in FIRE 2015 Shared Task on Mixed Script Information Retrieval. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015)*, pp. 53–56, Gandhinagar, India.

Jalam, R. (2003). *Apprentissage Automatique et Catégorisation de Textes Multilingues*. Ph.D. thesis, Université Lumière Lyon 2.

Jalam, R., & Teytaud, O. (2001a). Identification de la Langue et Catégorisation de Textes Basées sur les N-grammes. In Briand, H., & Guillet, F. (Eds.), *Journées Francophones d'extraction et de gestion de connaissances (EGC'2001)*, pp. 227–238, Nantes, France.

Jalam, R., & Teytaud, O. (2001b). Kernel-based Text Categorization. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'01)*, Vol. 3, pp. 1891–1896, Washington, DC, USA.

Jauhiainen, H., Jauhiainen, T., & Lindén, K. (2015). The Finno-Ugric Languages and The Internet Project. *Septentrio Conference Series*, *0*(2), 87–98.

Jauhiainen, T. (2010). Tekstin kielen automaattinen tunnistaminen. Master's thesis, University of Helsinki, Helsinki.

Jauhiainen, T., Jauhiainen, H., & Lindén, K. (2015a). Discriminating Similar Languages with Token-based Backoff. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 44–51, Hissar, Bulgaria.

Jauhiainen, T., Lindén, K., & Jauhiainen, H. (2015b). Language Set Identification in Noisy Synthetic Multilingual Documents. In *Proceedings of the Computational Linguistics and Intelligent Text Processing 16th International Conference, CICLing 2015*, pp. 633–643, Cairo, Egypt.

Jauhiainen, T., Lindén, K., & Jauhiainen, H. (2016). HeLI, a Word-Based Backoff Method for Language Identification. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 153–162, Osaka, Japan.

Jauhiainen, T., Lindén, K., & Jauhiainen, H. (2017a). Evaluating HeLI with Non-Linear Mappings. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 102–108, Valencia, Spain.

Jauhiainen, T., Lindén, K., & Jauhiainen, H. (2017b). Evaluation of Language Identification Methods Using 285 Languages. In *Proceedings of the 21st Nordic Conference on Computational Linguistics (NoDaLiDa 2017)*, pp. 183–191, Gothenburg, Sweden. Linköping University Electronic Press.

Jhamtani, H., Bhogi, S. K., & Raychoudhury, V. (2014). Word-level Language Identification in Bi-lingual Code-switched Texts. In *28th Pacific Asia Conference on Language, Information and Computation*, pp. 348–357, Phuket, Thailand.

Jiang, J. J., & Conrath, D. W. (1997). Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In *Proceedings of International Conference on Research on Computational Linguistics (ROCLING)*, Taiwan.

Jo, T. (2008). Neural Text Categorizer for Exclusive Text Categorization. *Journal of Infomration Processing Systems*, *4*, 77–86.

Jordan, M. I. (1986). Serial order: A Parallel Distributed Processing Approach. Tech. rep., Institute for Cognitive Science, University of California, San Diego.

Joshi, H., Bhatt, A., & Patel, H. (2013). Transliterated Search using Syllabification Approach. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2013)*, New Delhi, India.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2017). Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter*

of the Association for Computational Linguistics (EACL 2017), Vol. 2, pp. 427–431, Valencia, Spain.

Jourlin, P. (2017). Entity Recognition and Language Identification with FELTS. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Juola, P. (2006). Language Identification, Automatic. In *Encyclopedia of Language and Linguistics*, Vol. 6, pp. 508–––510. Elsevier, Amsterdam, Netherlands.

Jurgens, D., Tsvetkov, Y., & Jurafsky, D. (2017). Incorporating Dialectal Variability for Socially Equitable Language Identification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers)*, Vol. 2, pp. 51–57, Vancouver, Canada.

Kadri, S., & Moussaoui, A. (2013). An Effective Method to Recognize the Language of a Text in a Collection of Multilingual Documents. In *Proceedings of the International Conference on Electronics, Computer and Computation (ICECCO 2013)*, pp. 208–211, Ankara, Turkey.

Katz, S. M. (1987). Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *ASSP-35*(3), 400–401.

Kerwin, T. (2006). Classification of Natural Language Based on Character Frequency. Ohio Supercomputer Center.

Kheng, G., Léa, L., & Granitzer, M. (2017). INSA LYON and UNI PASSAU's participation at PAN@CLEF'17: Author Profiling task - Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Kikui, G.-I. (1996). Identifying the Coding System and Language of Online Documents on the Internet. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING '96)*, pp. 652–657, Copenhagen, Denmark.

Kim, S., & Park, J. (2007). Automatic Detection of Character Encoding and Language. Tech. rep., Stanford University.

King, B., & Abney, S. (2013). Labeling the Languages of Words in Mixed-Language Documents using Weakly Supervised Methods. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1110–1119, Atlanta, USA.

King, B., Radev, D., & Abney, S. (2014). Experiments in Sentence Language Identification with Groups of Similar Languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, pp. 146–154, Dublin, Ireland.

King, J., & Dehdari, J. (2008). An N-gram Based Language Identification System. The Ohio State University.

King, L., Baucom, E., Gilmanov, T., Kübler, S., Whyatt, D., Maier, W., & Rodrigues, P. (2014). The IUCL+ System: Word-Level Language Identification via Extended

Markov Models. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 102–106, Doha, Qatar.

King, L., Kübler, S., & Hooper, W. (2015). Word-level language identification in The Chymistry of Isaac Newton. *Digital Scholarship in the Humanities*, *30*(4), 532–540.

Kira, K., & Rendell, L. (1992). A practical approach to feature selection. In *Proceedings of the 9th International Machine Learning Conference*, pp. 249–256, San Mateo, USA.

Klein, D., Smarr, J., Nguyen, H., & Manning, C. D. (2003). Named Entity Recognition with Character-Level Models. In *Proceedings of the 7th Conference on Natural Language Learning*, pp. 180–183, Edmonton, Canada.

Kocmi, T., & Bojar, O. (2017). LanideNN: Multilingual Language Identification on Character Window. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Long Papers*, Vol. 1, pp. 927–936, Valencia, Spain.

Kodiyan, D., Hardegger, F., Neuhaus, S., & Cieliebak, M. (2017). Author Profiling with Bidirectional RNNs using Attention with GRUs - Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Kohavi, R. (1995). The Power of Decision Tables. In *Proceedings of the 8th European Conference on Machine Learning*, pp. 174–189, Crete, Greece. Springer.

Konstantopoulos, S. (2007). What's in a Name?. In *Proceedings of the 2007 Conference on Recent Advances in Natural Language Processing (RANLP-07)*, Borovets, Bulgaria.

Kralisch, A., & Mandl, T. (2006). Barriers to Information Access Across Languages on the Internet: Network and Language Effects. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, Vol. 3, p. 54b, Kauai, USA.

Kruengkrai, C., Srichaivattana, P., Sornlertlamvanich, V., & Isahara, H. (2005). Language identification based on string kernels. In *Proceedings of the 5th International Symposium on Communications and Information Technologies (ISCIT-2005)*, Vol. 2, pp. 896–899, Beijing, China.

Kulikowski, S. (1991). Language Identification of Short Texts. Newsgroup article, Educational Research and Development Center, The University of West Florida.

Kumar, R. V. R., Kumar, A. M., & Soman, K. (2015). AmritaCEN_NLP @ FIRE 2015 Language Identification for Indian Languages in Social Media Text. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015)*, pp. 28–30, Gandhinagar, India.

Laboreiro, G., Bošnjak, M., Sarmento, L., Rodrigues, E. M., & Oliveira, E. (2013). Determining Language Variant in Microblog Messages. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC'13)*, pp. 902–907, Coimbra, Portugal.

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pp. 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lamabam, P., & Chakma, K. (2016). A Language Identification System for Code-Mixed English-Manipuri Social Media Text. In *Proceedings of the IEEE International Conference on Engineering and Technology (ICETECH 2016)*, pp. 79–83, Coimbatore, Tamil Nadu, India.

Langer, S. (2001). Natural Languages and the World Wide Web. *Bulletin de Linguistique Appliquée et Générale (BULAG)*, *26*, 89–100.

Leidig, S. (2014). Single and Combined Features for the Detection of Anglicisms in German and Afrikaans. Bachelor's Thesis, Karlsruhe Institute of Technology.

Lewis, W. D., & Xia, F. (2010). Developing ODIN: A Multilingual Repository of Annotated Language Data for Hundreds of the World's Languages. *Literary and Linguistic Computing*, *25*(3), 303–319.

Lextek (2019). Lextek Language Identifier. Software available at `http://www.lextek.com/langid/li/`.

Li, H., Ma, B., & Lee, K. A. (2013). Spoken Language Recognition: From Fundamentals to Practice. *Proceedings of the IEEE*, *101*(5), 1136–1159.

Li, S., & Momoi, K. (2001). A Composite Approach to Language/Encoding Detection. In *Nineteenth International Unicode Conference (IUC19)*, San Jose, California, USA.

Li, Y., Cohn, T., & Baldwin, T. (2018). What's in a domain? learning domain-robust text representations using adversarial training. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics — Human Language Technologies (NAACL-HLT)*, pp. 474–479, New Orleans, USA.

Likasoft (2015). Polyglot 3000. Software available at `http://www.polyglot3000.com`.

Lin, C.-C., Ammar, W., Levin, L., & Dyer, C. (2014). The CMU Submission for the Shared Task on Language Identification in Code-Switched Data. In *Proceedings of First Workshop on Computational Approaches to Code Switching*, pp. 80–86, Doha, Qatar.

Lin, D. (1998). An Information-Theoretic Definition of Similarity. In Shavlik, J. W. (Ed.), *Proceedings of the 15th International Conference on Machine Learning (ICML-98)*, pp. 296–304, Madison, Wisconsin, USA.

Linde, Y., Buzo, A., & Gray, R. M. (1980). An Algorithm for Vector Quantizer Design. *IEEE Transactions on Communications*, *COM-28*(1), 84–95.

Ling, W., Xiang, G., Dyer, C., Black, A., & Trancoso, I. (2013). Microblogs as Parallel Corpora. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 176–186, Sofia, Bulgaria.

Lins, R. D., & Gonçalves, P. J. (2004). Automatic Language Identification of Written Texts. In *Proceedings of the 2004 ACM symposium on Applied Computing (SAC 2004)*, pp. 1128–1133, Nicosia, Cyprus.

Littlestone, N. (1987). Learning Quickly When Irrelevant Attributes Abound: A New Linear-threshold Algorithm. *Machine learning*, *2*(4), 285–318.

Ljubešić, N., & Klubička, F. (2014). {bs,hr,sr}WaC – Web Corpora of Bosnian, Croatian and Serbian. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, pp. 29–35, Gothenburg, Sweden. Association for Computational Linguistics.

Ljubešić, N., & Kranjcić, D. (2014). Discriminating between VERY Similar Languages among Twitter Users. In *Proceedings of the 9th Language Technologies Conference*, pp. 90–94, Ljubljana, Slovenia.

Ljubešić, N., & Kranjcić, D. (2015). Discriminating Between Closely Related Languages on Twitter. *Informatica*, *39*.

Ljubešić, N., Mikelić, N., & Boras, D. (2007). Language Identification: How to Distinguish Similar Languages?. In *Proceedings of the 29th International Conference on Information Technology Interfaces (ITI 2007)*, pp. 541–546, Cavtat/Dubrovnik, Croatia.

Lu, M., & Mohamed, M. (2011). LAHGA: Arabic Dialect Classifier. Unpublished Report...

Ludovik, Y., & Zacharski, R. (1999). Multilingual Document Language Recognition for Creating Corpora. Tech. rep., New Mexico State University.

Lui, M. (2011). *langid.py*. Software available at `https://github.com/saffsd/langid.py` (last updated on July 2017).

Lui, M. (2014a). *Generalized Language Identification*. Ph.D. thesis, The University of Melbourne.

Lui, M. (2014b). *langid.c*. Software available at `https://github.com/saffsd/langid.c` (last updated on September 2017).

Lui, M. (2014c). *langid.js*. Software available at `https://github.com/saffsd/langid.js` (last updated on July 2014).

Lui, M., & Baldwin, T. (2011). Cross-domain Feature Selection for Language Identification. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, Vol. 1, pp. 553–561, Chiang Mai, Thailand.

Lui, M., & Baldwin, T. (2012). langid.py: An Off-the-shelf Language Identification Tool. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012) Demo Session*, pp. 25–30, Jeju, Republic of Korea.

Lui, M., & Baldwin, T. (2014). Accurate Language Identification of Twitter Messages. In *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, pp. 17–25, Gothenburg, Sweden. Association for Computational Linguistics.

Lui, M., & Cook, P. (2013). Classifying English Documents by National Dialect. In *Proceedings of Australasian Language Technology Association Workshop 2013 (ALTA 2013)*, pp. 5–15, Brisbane, Australia.

Lui, M., Lau, J. H., & Baldwin, T. (2014a). Automatic Detection and Language Identification of Multilingual Documents. *Transactions of the Association for Computational Linguistics*, *2*, 27–40.

Lui, M., Letcher, N., Adams, O., Duong, L., Cook, P., & Baldwin, T. (2014b). Exploring Methods and Resources for Discriminating Similar Languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, pp. 129–138, Dublin, Ireland.

MacNamara, S., Cunningham, P., & Byrne, J. (1998). Neural Networks for Language Identification: a Comparative Study. *Information processing and management*, *34*(4), 395–403.

Maier, W., & Gómez-Rodríguez, C. (2014). Language Variety Identification in Spanish Tweets. In *Proceedings of the EMNLP'2014 Workshop on Language Technology for Closely Related Languages and Language Variants (LT4CloseLang 2014)*, pp. 25–35, Doha, Qatar. Association for Computational Linguistics.

Majliš, M. (2011). Large Multilingual Corpus. Master's thesis, Charles University in Prague, Prague.

Majliš, M. (2012a). YALI (Yet Another Language Identifier).. Software available at `https://github.com/martin-majlis/YALI` (last updated on Feb 2019).

Majliš, M. (2012b). Yet Another Language Identifier. In *Proceedings of the Student Research Workshop at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 46–54, Avignon, France.

Malmasi, S. (2017). Open-Set Language Identification. *arXiv preprint*, *arXiv:1707.04817*.

Malmasi, S., & Dras, M. (2015a). Automatic Language Identification for Persian and Dari Texts. In *Proceedings of the 14th Conference of the Pacific Association for Computational Linguistics, PACLING' 15*, pp. 59–64, Bali, Indonesia.

Malmasi, S., & Dras, M. (2015b). Language Identification using Classifier Ensembles. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 35–43, Hissar, Bulgaria.

Malmasi, S., & Dras, M. (2017). Feature Hashing for Language and Dialect Identification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pp. 399–403, Vancouver, Canada.

Malmasi, S., Refaee, E., & Dras, M. (2015). Arabic Dialect Identification using a Parallel Multidialectal Corpus. In *Proceedings of the 14th Conference of the Pacific Association for Computational Linguistics, PACLING' 15*, pp. 209–217, Bali, Indonesia.

Malmasi, S., & Zampieri, M. (2016). Arabic Dialect Identification in Speech Transcripts. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 106–113, Osaka, Japan.

Malmasi, S., & Zampieri, M. (2017a). Arabic Dialect Identification Using iVectors and ASR Transcripts. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 178–183, Valencia, Spain.

Malmasi, S., & Zampieri, M. (2017b). German Dialect Identification in Interview Transcriptions. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 164–169, Valencia, Spain.

Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A., & Tiedemann, J. (2016). Discriminating Between Similar Languages and Arabic Dialect Identification: A Report on the Third DSL Shared Task. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 1–14, Osaka, Japan.

Mandal, S., Banerjee, S., Naskar, S. K., Rosso, P., & Bandyopadhyay, S. (2015). Adaptive Voting in Multiple Classifier Systems for Word Level Language Identification. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015)*, pp. 49–52, Gandhinagar, India.

Mandl, T., Shramko, M., Tartakovski, O., & Womser-Hacker, C. (2006). Language Identification in Multi-lingual Web-Documents. In *Proceedings of the 11th International Conference on Applications of Natural Language to Information Systems (NLDB 2006)*, pp. 153–163, Klagenfurt, Austria.

Marcadet, J.-C., Fischer, V., & Waast-Richard, C. (2005). A Transformation-based Learning Approach to Language Identification for Mixed-lingual Text-to-speech Synthesis. In *Proceedings of the 6th Interspeech and 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, pp. 2248–2251, Lisbon, Portugal.

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, *19*(2), 313–330.

Markov, I., Gómez-Adorno, H., & Sidorov, G. (2017). Language- and Subtask-Dependent Feature Selection and Classifier Parameter Tuning for Author Profiling—Notebook for PAN at CLEF 2017. In *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland.

Martadinata, P., Trisedya, B. D., Manurung, H. M., & Adriani, M. (2016). Building Indonesian Local Language Detection Tools Using Wikipedia Data. In Murakami, Y., & Lin, D. (Eds.), *Worldwide Language Service Infrastructure*, pp. 113–123. Springer.

Martinc, M., Škrjanec, I., Zupan, K., & Pollak, S. (2017). PAN 2017: Author Profiling - Gender and Language Variety Prediction—Notebook for PAN at CLEF 2017. In *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland.

Martins, B., & Silva, M. J. (2005). Language Identification in Web Pages. In *Proceedings of the 2005 ACM symposium on Applied computing*, pp. 764–768, Santa Fe, USA.

Mather, L. A. (1998). A Linear Algebra Approach to Language Identification. In Munson, E. V. (Ed.), *Proceedings of the 4th International Workshop Principles of Digital Document Processing (PODDP' 98)*, pp. 92–103, Saint Malo, France.

Mathur, P., Misra, A., & Budur, E. (2017). LIDE: Language Identification from Text Documents. *arXiv preprint*, *arXiv:1701.03682*.

Mayer, U. F. (2012). Bootstrapped Language Identification for Multi-site Internet Domains. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2012)*, pp. 579–585, Beijing, China.

McNamee, P. (2005). Language Identification: A Solved Problem Suitable for Undergraduate Instruction. *Journal of Computing Sciences in Colleges*, *20*(3), 94–101.

McNamee, P. (2016). Language and Dialect Discrimination Using Compression-Inspired Language Models. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 195–203, Osaka, Japan.

Medvedeva, M., Kroon, M., & Plank, B. (2017). When Sparse Traditional Models Outperform Dense Neural Networks: the Curious Case of Discriminating between Similar Languages. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 156–163, Valencia, Spain.

Mendizabal, I., Carandell, J., & Horowitz, D. (2014). TweetSafa: Tweet Language Identification. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located*

with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN), pp. 21–25, Girona, Spain.

Mendoza, I., & Mendelsohn, J. (2017). Exploring Techniques in Distinguishing Similar Languages. Stanford course project.

Mikolov, T., Chen, K., Corrado, G., & Jeffrey, D. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs.CL].

Minocha, A., & Tyers, F. M. (2014). Subsegmental Language Detection in Celtic Language Text. In *Proceedings of the First Celtic Language Techonology Workshop (CLTW 2014)*, pp. 76–80, Dublin, Ireland.

Miura, Y., Taniguchi, T., Taniguchi, M., Misawa, S., & Ohkuma, T. (2017). Using Social Networks to Improve Language Variety Identification with Neural Networks. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*, pp. 263–270, Taipei, Taiwan.

Mokhov, S. A. (2010a). A MARF Approach to DEFT 2010. In *Proceedings of the 6th DEFT Workshop (DEFT' 10)*, pp. 35–49.

Mokhov, S. A. (2010b). Complete Complimentary Results Report of the MARF's NLP Approach to the DEFT 2010 Competition. *CoRR, abs/1006.3787*.

Molina, G., Rey-Villamizar, N., Solorio, T., AlGhamdi, F., Ghoneim, M., Hawwari, A., & Diab, M. (2016). Overview for the Second Shared Task on Language Identification in Code-Switched Data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 40–49, Austin, TX, USA.

Moodley, A. (2016). Language Identification With Decision Trees: Identification Of Individual Words In The South African Languages. Bachelor's Thesis, University of South Africa.

Mukherjee, A., Ravi, A., & Datta, K. (2014). Mixed-script Query Labelling Using Supervised Learning and Ad Hoc Retrieval Using Sub Word Indexing. In *FIRE' 14 Proceedings of the Forum for Information Retrieval*, pp. 86–90, Bangalore, India.

Murthy, K. N., & Kumar, G. B. (2006). Language Identification from Small Text Samples. *Journal of Quantitative Linguistics*, *13*(1), 57–80.

Mustonen, S. (1965). Multiple Discriminant Analysis in Linguistic Problems. *Statistical Methods in Linguistics*, *4*, 37–44.

Muthusamy, Y. K., & Spitz, A. L. (1997). Automatic Language Identification. In Cole, R., Mariani, J., Uszkoreit, H., Varile, G. B., Zaenen, A., Zampolli, A., & Zue, V. (Eds.), *Web Edition: Survey of the State of the Art in Human Language Technology*, pp. 314–317. Cambridge University Press, Cambridge, UK.

Nakamura, Y. (1971). Identification of Languages with Short Sample Texts – A Linguometric Study. *Library and information science*, *9*, 459–481.

Nakatani, S. (2010). Language Detection Library for Java. Available at: `http://code.google.com/p/language-detection/` (last updated on March 2014).

Nakatani, S. (2011). ldig (Language Detection with Infinity Gram). Available at: `https://github.com/shuyo/ldig` (last updated on July 2013).

Ney, H., Essen, U., & Kneser, R. (1994). On structuring probabilistic dependences in stochastic language modelling. *Computer Speech and Language*, *8*(1), 1–38.

Ng, A. Y., & Jordan, M. I. (2002). On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes. In Becker, S., Thrun, S., & Obermayer, K. (Eds.), *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pp. 841–848, Vancouver, British Columbia, Canada.

Ng, C.-C., & Selamat, A. (2009). Improved Letter Weighting Feature Selection on Arabic Script Language Identification. In *Proceedings of the 1st Asian Conference on Intelligent Information and Database Systems (ACIIDS 2009)*, pp. 150–154, Dong Hoi, Vietnam. IEEE.

Nguyen, D., & Cornips, L. (2016). Automatic Detection of Intra-Word Code-Switching. In *Proceedings of the 14th Annual SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 82–86, Berlin, Germany.

Nguyen, D., & Dogruöz, A. S. (2013). Word Level Language Identification in Online Multilingual Communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pp. 857–862, Seattle, USA.

Nobesawa, S., & Tahara, I. (2005). Language Identification for Person Names Based on Statistical Information. In *Proceedings of the 19th Pacific Asia Conference on Language, Information and Computation (PACLIC 19)*, No. 1027 in Y05, Taipei, Taiwan.

Noh, N. M., Talib, M. R. A., Ahmad, A., Halim, S. A., & Mohamed, A. (2009). Malay Language Document Identification Using BPNN. In Mastorakis, N. E., Croitoru, A., Balas, V. E., Son, E., & Mladenov, V. (Eds.), *Proceedings of the 10th WSEAS international conference on Neural networks*, pp. 163–168, Prague, Czech Republic.

Okanohara, D., & Tsujii, J. (2009). Text Categorization with All Substring Features. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pp. 838–846, Miami, USA.

Oliveira, R. R., & Neto, R. F. d. O. (2017). Using Character N-grams and Style Features for Gender and Language Variety Classification – Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Ozbek, G., Rosenn, I., & Yeh, E. (2006). Language Classification in Multilingual Documents. Tech. rep., Stanford University.

Panich, L. (2015). Comparison of Language Identification Techniques. Bachelor's Thesis, Heinrich Heine Universität Düsseldorf.

Papalexakis, E. E., Nguyen, D., & Dogruöz, A. S. (2014). Predicting Code-Switching in Multilingual Communication for Immigrant Communities. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 42–50, Doha, Qatar.

Pasha, A., Al-Badrashiny, M., Diab, M., El Kholy, A., Eskander, R., Habash, N., Pooleery, M., Rambow, O., & Roth, R. M. (2014). MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In *Proceedings of The Ninth International Conference on Language Resources and Evaluation (LREC 2014)*, pp. 1094–1101, Reykjavik, Iceland.

Pavan, K., Tandon, N., & Varma, V. (2010). Addressing Challenges in Automatic Language Identification of Romanized Text. In *Proceedings of 8th International Conference on Natural Language Processing (ICON-2010)*, Kharagpur, India.

Pawelka, T., & Jürgens, E. (2015). Is This Code Written in English? A Study of the Natural Language of Comments and Identifiers in Practice. In *Proceedings of the 31st International Conference on Software Maintenance and Evolution (ICSME)*, Bremen, Germany.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*(Oct), 2825–2830.

Peirsman, Y., Geeraerts, D., & Speelman, D. (2010). The Automatic Identification of Lexical Variation Between Language Varieties. *Natural Language Engineering*, *16*(04), 469–491.

Peng, F., Feng, F., & McCallum, A. (2004). Chinese Segmentation and New Word Detection Using Conditional Random Fields. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pp. 562–568, Geneva, Switzerland.

Peng, F., & Schuurmans, D. (2003). Combining Naive Bayes and n-Gram Language Models for Text Classification. In *Proceedings of the 25th European Conference on IR Research, Advances in Information Retrieval: (ECIR 2003)*, pp. 335–350, Pisa, Italy. Springer Berlin Heidelberg.

Pethö, G., & Mózes, E. (2014). An N-gram-based Language Identification Algorithm for Variable-length and Variable-language Texts. *Argumentum*, *10*, 56–82.

Pham, T., & Tran, D. (2003). VQ-based Written Language Identification. In *Proceedings of the Seventh International Symposium on Signal Processing and Its Applications (ISSPA 2003)*, Vol. 1, pp. 513–516, Paris, France.

Pienaar, W., & Snyman, D. (2010). Spelling Checker-based Language Identification for the Eleven Official South African Languages. In *Proceedings of the 21st Annual Symposium of the Pattern Recognition Association of South Africa*, pp. 213–217, Stellenbosch, South Africa.

Piergallini, M., Shirvani, R., Gautam, G. S., & Chouikha, M. (2016a). The Howard University System Submission for the Shared Task in Language Identification in Spanish-English Codeswitching. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 116–120, Austin, TX, USA.

Piergallini, M., Shirvani, R., Gautam, G. S., & Chouikha, M. (2016b). Word-Level Language Identification and Predicting Codeswitching Points in Swahili-English Language Data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 21–29, Austin, TX, USA.

Pla, F., & Hurtado, L.-F. (2015). Language Identification in Twitter: A Study Case of Multiclass and Multilabel Text Classification Problem. *International Journal of Computational Linguistics and Applications*, *6*(1), 135–150.

Pla, F., & Hurtado, L.-F. (2017). Language Identification of Multilingual Posts from Twitter: A Case Study. *Knowledge and Information Systems*, *51*(3), 965–989.

Plaza Cagigós, S. (2017). Catdetect, a framework for detecting Catalan tweets. Bachelor thesis, Universitat de Lleida.

Popatov, S. (2016). *whatlang-rs*. Software available at `https://github.com/greyblake/whatlang-rs` (last updated May 2019).

Porta, J. (2014). Twitter Language Identification using Rational Kernels and its Potential Application to Sociolinguistics. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, pp. 17–20, Girona, Spain.

Porta, J., & Sancho, J.-L. (2014). Using Maximum Entropy Models to Discriminate between Similar Languages and Varieties. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, pp. 120–128, Dublin, Ireland.

Poulston, A., Waseem, Z., & Stevenson, M. (2017). Using TF-ID n-gram and Word Embedding Cluster Ensembles for Author Profiling – Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Poutsma, A. (2002). Applying Monte Carlo Techniques to Language Identification. *Language and Computers*, *45*(1), 179–189.

Prager, J. M. (1999). Linguini: Language Identification for Multilingual Documents. In *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences (HICSS-32)*, Maui, USA.

Qafmolla, M. A. N. (2017). Automatic Language Identification. *European Journal of Language and Literature Studies*, *7*(1), 140–150.

Qiao, S., & Lévy, D. (2015). Similar Language Detection. Stanford University.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann, San Francisco, CA, USA.

Rabiner, L. R., & Juang, B. H. (1986). An Introduction to Hidden Markov Models. *IEEE ASSP Magazine*, *3*(1), 4–16.

Radford, W., & Gallé, M. (2016). Discriminating Between Similar Languages in Twitter Using Label Propagation. *arXiv preprint*, *arXiv:1607.05408*.

Rafidha Rehiman, K. A., Keerthy, A. S., Lakshmi, K. S., & Sreekumar, A. (2013). A Language Identification and Conversion System for Malayalam to Ensure Security. In *3rd National Conference on Indian Language Computing (NCILC 2013)*, Cochin, Kerala, India.

Raghavi, K. C., Chinnakotla, M., & Shrivastava, M. (2015). "Answer ka type kya he?" Learning to Classify Questions in Code-Mixed Language. In *WWW' 15 Companion Proceedings of the 24th International Conference on World Wide Web*, pp. 853–858, Florence, Italy.

Raj, A., & Karfa, S. (2014). A List-searching Based Approach for Language Identification in Bilingual Text: Shared Task Report by Asterisk. In *Working Notes of the Shared*

Task on Transliterated Search at Forum for Information Retrieval Evaluation FIRE' 14, Bangalore, India.

Ramisch, C. (2008). N-gram Models for Language Detection. Technical Report.

Ranaivo-Malançon, B. (2006). Automatic Identification of Close Languages – Case study: Malay and Indonesian. *ECTI Transactions on Computer and Information Technology*, *2*(2), 126–134.

Ranaivo-Malançon, B., & Ng, P. K. (2005). Language Identifier for Bahasa Malaysia and Bahasa Indonesia. In *Proceedings of the 1st Malaysian Software Engineering Conference (MySEC'05)*, pp. 257–259, Penang, Maylaysia.

Rangel, F., Franco-Salvador, M., & Rosso, P. (2017a). A Low Dimensionality Representation for Language Variety Identification. In *Proceedings of the 17th International Conference on Computational Linguistics and Intelligent Text Processing (CICLING)*.

Rangel, F., Rosso, P., Potthast, M., & Stein, B. (2017b). Overview of the 5th Author Profiling Task at PAN 2017: Gender and Language Variety Identification in Twitter. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Ranjan, P., Raja, B., Priyadharshini, R., & Balabantaray, R. C. (2016). A Comparative Study on Code-Mixed Data of Indian Social Media vs Formal Text. In Niranjan, S. K., & Aradhya, V. N. M. (Eds.), *Proceedings of the 2nd International Conference on Contemporary Computing and Informatics (IC3I 2016)*, pp. 608–611, Noida, India. IEEE.

Rau, M. D. (1974). Language Identification by Statistical Analysis. Master's thesis, Naval Postgraduate School, Monterey.

Rehůřek, R., & Kolkus, M. (2009). Language Identification on the Web: Extending the Dictionary Method. In *Proceedings of the 10th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2009)*, pp. 357–368, Mexico City, Mexico.

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, *62*, 107–136.

Rijhwani, S., Sequeira, R., Choudhury, M., Bali, K., & Maddila, C. S. (2017). Estimating Code-Switching on Twitter with a Novel Generalized Word-Level Language Detection Technique. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1971–1982, Vancouver, Canada.

Rodrigues, P. (2012). *Processing Highly Variant Language Using Incremental Model Selection.* Ph.D. thesis, Indiana University.

Rodríguez, J. J., Kuncheva, L. I., & Alonso, C. J. (2006). Rotation Forest: A New Classifier Ensemble Method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(10), 1619–1630.

Romsdorfer, H., & Pfister, B. (2007). Text Analysis and Language Identification for Polyglot Text-to-speech Synthesis. *Speech communication*, *49*(9), 697–724.

Rosner, M., & Farrugia, P.-J. (2007). A Tagging Algorithm for Mixed Language Identification in a Noisy Domain. In *INTERSPEECH-2007, 8th Annual Conference of the International Speech Communication Association*, pp. 190–193, Antwerp, Belgium.

Rowe, N. C., Schwamm, R., & Garfinkel, S. L. (2013). Language Translation for File Paths. *Digital Investigation*, *10*.

Roy, R. S., Choudhury, M., Majumder, P., & Agarwal, K. (2013). Overview of the FIRE 2013 Track on Transliterated Search. In Majumder, P., Mitra, M., Agrawal, M., & Mehta, P. (Eds.), *Proceedings of the 5th Forum on Information Retrieval Evaluation (FIRE '13)*, New Delhi, India. ACM.

Rubinstein, Y. D., & Hastie, T. (1997). Discriminative vs Informative Learning. In Heckerman, D., Mannila, H., Pregilbon, D., & Uthurusamy, R. (Eds.), *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, pp. 49–53, Newport Beach, California, USA.

Russell, G., & Lapalme, G. (2003). Automatic Identification of Language and Encoding. Tech. rep., Laboratoire de Recherche Appliquée en Linguistique Informatique (RALI), Université de Montréal.

Sadat, F., Kazemi, F., & Farzindar, A. (2014a). Automatic Identification of Arabic Dialects in Social Media. In *Proceedings of the first international workshop on Social media retrieval and analysis (SoMeRA 2014)*, pp. 35–40, Gold Coast, QLD, Australia. ACM.

Sadat, F., Kazemi, F., & Farzindar, A. (2014b). Automatic Identification of Arabic Language Varieties and Dialects in Social Media. In Lin, S.-d., Ku, L.-W., Cambria, E., & Kuo, T.-T. (Eds.), *Proceedings of the Second Workshop on Natural Language Processing for Social Media (SocialNLP 2014)*, pp. 22–27, Dublin, Ireland.

Saharia, N. (2017). Phone-based Identification of Language in Code-mixed Social Network Data. *Journal of Statistics and Management Systems*, *20*(4), 565–574.

Salloum, W., Elfardy, H., Alamir-Salloum, L., Habash, N., & Diab, M. (2014). Sentence Level Dialect Identification for Machine Translation System Selection. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 772–778, Baltimore, USA.

Samih, Y. (2017). *Dialectal Arabic Processing Using Deep Learning*. Ph.D. thesis, Heinrich-Heine-Universität Düsseldorf, Düsseldorf, Germany.

Samih, Y., Maharjan, S., Attia, M., Kallmeyer, L., & Solorio, T. (2016). Multilingual Code-switching Identification via LSTM Recurrent Neural Networks. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 50–59, Austin, TX, USA.

Samih, Y., & Maier, W. (2016). Detecting Code-Switching in Moroccan Arabic Social Media. In *Proceedings of the 4th International Workshop on Natural Language Processing for Social Media (SocialNLP 2016 IJCAI)*, New York City, USA.

Sanchez-Perez, M. A., Markov, I., Gómez-Adorno, H., & Grigori, S. (2017). Comparison of Character N-grams and Lexical Features on Author, Gender, and Language Variety Identification on the Same Spanish News Corpus. In Jones, G. J. F., Lawless, S.,

Gonzalo, J., Kelly, L., Goeuriot, L., Mandl, T., Cappellato, L., & Ferro, N. (Eds.), *Proceedings of the 8th International Conference of the CLEF Association (CLEF 2017)*, pp. 145–151, Dublin, Ireland. Springer.

Sangeeth, K. (2017). *whatthelang*. Software available at `https://github.com/indix/whatthelang` (last updated on November 2017).

Scannell, K. P. (2007). The Crúbadán Project: Corpus Building for Under-resourced Languages. In *Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop*, pp. 5–15, Louvain-la-Neuve, Belgium.

Schaetti, N. (2017). UniNE at CLEF 2017: TF-IDF and Deep-Learning for Author Profiling - Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Scheelen, F. (2003). *libtextcat*. Software available at `http://software.wise-guys.nl/libtextcat/`.

Scherrer, Y., & Rambow, O. (2010). Word-based Dialect Identification with Georeferenced Rules. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pp. 1151–1161, Massachusetts, USA. Association for Computational Linguistics.

Schulz, S., & Keller, M. (2016). Code-Switching Ubique Est - Language Identification and Part-of-Speech Tagging for Historical Mixed Text. In *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH 2016)*, pp. 43–51, Berlin, Germany.

Sebastiani, F. (2002). Machine Learning in Automated Text Categorization. *ACM Computing Surveys (CSUR)*, *34*(1), 1–47.

Seifart, F., & Mundry, R. (2015). Quantitative Comparative Linguistics based on Tiny Corpora: N-gram Language Identification of Wordlists of Known and Unknown Languages from Amazonia and Beyond. *Journal of Quantitative Linguistics*, *22*(3).

Selamat, A., & Akosu, N. (2016). Word-length Algorithm for Language Identification of Under-resourced Languages. *Journal of King Saud University - Computer and Information Sciences*, *28*, 457–469.

Selamat, A., & Ng, C.-C. (2008). Arabic Script Documents Language Identifications Using Fuzzy ART. In Al-Dabass, D., Turner, S., Tan, G., & Abraham, A. (Eds.), *Proceedings of the Second Asia International Conference on Modeling & Simulation (AMS 2008)*, pp. 528–533, Kuala Lumpur, Malaysia.

Selamat, A., Ng, C.-C., & Mikami, Y. (2007). Arabic Script Web Documents Language Identification Using Decision Tree-ARTMAP Model. In *Proceedings of the International Conference on Convergence Information Technology (ICCIT 2007)*, pp. 717–722, Gyeongju, Korea. IEEE.

Sequeira, R., Choudhury, M., Gupta, P., Rosso, P., Kumar, S., Banerjee, S., Naskar, S. K., Bandyopadhyay, S., Chittaranjan, G., Das, A., & Chakma, K. (2015). Overview of FIRE-2015 Shared Task on Mixed Script Information Retrieval. In *Working Notes of*

the Forum for Information Retrieval Evaluation (FIRE 2015), pp. 21–27, Gandhinagar, India.

Shah, S., Jain, V., Jain, S., Mittal, A., Verma, J., Tripathi, S., & Kumar, D. R. (2015). Hierarchical classification for Multilingual Language Identification and Named Entity Recognition. In *Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015)*, pp. 21–27, Gandhinagar, India.

Shashirekha, H. L. (2014). Automatic Language Identification from Written Texts - An Overview. *International Journal of Innovative Research in Computer and Communication Engineering*, *2*(5), 156–160.

Shiells, K., & Pham, P. (2010). Unsupervised Clustering for Language Identification. Project Report, Stanford University.

Shrestha, P. (2014). Incremental N-gram Approach for Language Identification in Code-Switched Text. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 133–138, Doha, Qatar.

Shrestha, P. (2016). Codeswitching Detection via Lexical Features using Conditional Random Fields. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 121–126, Austin, TX, USA.

Sibun, P., & Reynar, J. C. (1996). Language Identification: Examining the Issues. In *Proceedings of the 5th Annual Symposium on Document Analysis and Information Retrieval (SDAIR-96)*, pp. 125–135, Las Vegas, USA.

Sibun, P., & Spitz, A. L. (1994). Language Determination: Natural Language Processing from Scanned Document Images. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, pp. 15–21, Stuttgart, Germany.

Sierra, S., Montes-y Gómez, M., Solorio, T., & González, F. A. (2017). Convolutional Neural Networks for Author Profiling—Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Sikdar, U. K., & Gambäck, B. (2016). Language Identification in Code-Switched Text Using Conditional Random Fields and Babelnet. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 127–131, Austin, TX, USA.

Simaki, V., Simakis, P., Paradis, C., & Kerren, A. (2017). Identifying the Authors' National Variety of English in Social Media Texts. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP 2017)*, pp. 671–678, Varna, Bulgaria. INCOMA Ltd.

Simões, A., Almeida, J. J., & Byers, S. D. (2014). Language Identification: a Neural Network Approach. In Pereira, M. J. V., Leal, J. P., & Simões, A. (Eds.), *Proceedings of the 3rd Symposium on Languages, Applications and Technologies (SLATE 2014)*, pp. 251–265, Bragança, Portugal.

Simons, G. F., & Fennig, C. D. (Eds.). (2017). *Ethnologue: Languages of the World, Twentieth Edition*. SIL International, Dallas, USA. Online version: `http://www.ethnologue.com`.

Singh, A. K. (2006). Study of Some Distance Measures for Language and Encoding Identification. In *Proceedings of the Workshop on Linguistic Distances*, pp. 63–72, Sydney, Australia.

Singh, A. K. (2010). *Modeling and Application of Linguistic Similarity*. Ph.D. thesis, International Institute of Information Technology, Hyderabad.

Singh, A. K., & Gorla, J. (2007). Identification of Languages and Encodings in a Multilingual Document. In *Proceedings of the 3rd ACL SIGWAC Workshop on Web As Corpus (WAC3-2007)*, pp. 95–108, Louvain-la-Neuve, Belgium.

Singh, A. K., & Goyal, P. (2014). A Language Identification Method Applied to Twitter Data. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, pp. 26–29, Girona, Spain.

Sinha, N., & Srinivasa, G. (2014). Hindi-English Language Identification, Named Entity Recognition and Back Transliteration: Shared Task System Description. In *Working Notes os Shared Task on Transliterated Search at Forum for Information Retrieval Evaluation FIRE' 14*, Bangalore, India.

Sitaram, S., Chandu, K. R., Rallabandi, S. K., & Black, A. W. (2019). A Survey of Code-switched Speech and Language Processing. *arXiv preprint*, *arXiv:1904.00784v2*.

Sites, D. (2013). Compact language detector 2. Software available at `https://github.com/CLD2Owners/cld2` (last updated on August 2015).

Solorio, T., Blair, E., Maharjan, S., Bethard, S., Diab, M., Gohneim, M., Hawwari, A., AlGhamdi, F., Hirschberg, J., Chang, A., & Fung, P. (2014). Overview for the First Shared Task on Language Identification in Code-Switched Data. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pp. 62–72, Doha, Qatar.

Souter, C., Churcher, G., Hayes, J., Hughes, J., & Johnson, S. (1994). Natural Language Identification using Corpus-Based Models. *Hermes, Journal of Linguistics*, *13*, 183–203.

Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., Tufis, D., & Varga, D. (2006). The JRC-Acquis: A Multilingual Aligned Parallel Corpus with 20+ Languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*, pp. 2142–2147, Geona, Italy.

Stensby, A., Oommen, B. J., & Granmo, O.-C. (2010). Language Detection and Tracking in Multilingual Documents Using Weak Estimators. In *Proceedings of the Joint IAPR International Workshop Structural, Syntactic, and Statistical Pattern Recognition (SSPR&SPR 2010)*, pp. 600–609, Cesme, Izmir, Turkey.

Sterneberg, E. (2012). Language Identification of Person Names Using Cascaded SVMs. Bachelor's Thesis, Uppsala University, Uppsala.

Stolcke, A. (2002). SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP-2002)*, pp. 901–904, Denver, Colorado, USA.

Stupar, M., Jurić, T., & Ljubešić, N. (2011). Language Identification of Web Data for Building Linguistic Corpora. In *Proceedings of the 3rd International Conference on The Future of Information Sciences (INFuture 2011)*, pp. 365–372, Zagreb, Croatia.

Suzuki, I., Mikami, Y., Ohsato, A., & Chubachi, Y. (2002). A Language and Character Set Determination Method Based on n-gram Statistics. *ACM Transactions on Asian Language Information Processing (TALIP)*, *1*(3), 269–278.

Takçı, H., & Ekinci, E. (2012). Minimal Feature Set in Language Identification and Finding Suitable Classification Method with it. *Procedia Technology*, *1*, 444–448.

Takçı, H., & Güngör, T. (2012). A High Performance Centroid-based Classification Approach for Language Identification. *Pattern Recognition Letters*, *33*(16), 2077–2084.

Tan, L., Zampieri, M., Ljubešić, N., & Tiedemann, J. (2014). Merging Comparable Data Sources for the Discrimination of Similar Languages: The DSL Corpus Collection. In *Proceedings of the 7th Workshop on Building and Using Comparable Corpora (BUCC)*, Reykjavik, Iceland.

Teahan, W. J. (2000). Text Classification and Segmentation Using Minimum Cross-Entropy. In *Proceedings of the 6th International Conference Recherche d' Information Assistee par Ordinateur (RIAO' 00)*, pp. 943–961, Paris, France.

Tellez, E. S., Miranda-Jiménez, S., Graff, M., & Moctezuma, D. (2017). Gender and Language-variety Identification with MicroTC – Notebook for PAN at CLEF 2017. In Cappellato, L., Ferro, N., Goeuriot, L., & Mandl, T. (Eds.), *Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop*, Dublin, Ireland. CEUR-WS.org.

Thomas, S., & Verma, A. (2007). Language Identification of Person Names using CF-IOF based Weighing Function. In *8th Annual Conference of the International Speech Communication Association (INTERSPEECH 2007)*, pp. 1769–1772, Antwerp, Belgium.

Tian, J., Häkkinen, J., Riis, S., & Jensen, K. J. (2002). On Text-based Language Identification for Multilingual Speech Recognition Systems. In *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP2002)*, Denver, Colorado, USA.

Tian, J., & Suontausta, J. (2003). Scalable Neural Network Based Language Identification from Written Text. In *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*, Vol. 1, pp. 48–51, Hong Kong.

Tiedemann, J. (2012). Parallel Data, Tools and Interfaces in OPUS. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, pp. 2214–2218, Istanbul, Turkey.

Tiedemann, J., & Ljubešić, N. (2012). Efficient Discrimination Between Closely Related Languages. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pp. 2619–2634, Mumbai, India.

Tillmann, C., Al-Onaizan, Y., & Mansour, S. (2014). Improved Sentence-Level Arabic Dialect Classification. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, pp. 110–119, Dublin, Ireland.

Tomović, A., & Janičić, P. (2007). A Variant of N-Gram Based Language Classification. In *Proceedings of the 10th Congress of the Italian Association for Artificial Intelligence Artificial Intelligence and Human-Oriented Computing (AI* IA 2007)*, pp. 410–421, Rome, Italy.

Tran, D., & Sharma, D. (2005). Markov Models for Written Language Identification. In *Proceedings of the 12th International Conference on Neural Information Processing*, pp. 67–70, Taipei, Taiwan.

Tran, G. B., Nguyen, D. B., & Kieu, B. T. (2010). *n*-gram based approach for multilingual language identification. Poster. available at `http://comp.mq.edu.au/programming/task_description/VILangTek.pdf`..

Tratz, S., Briesch, D., Laoudi, J., & Voss, C. (2013). Tweet Conversation Annotation Tool with a Focus on an Arabic Dialect, Moroccan Darija. In *Proceedings of the 7th Linguistic Annotation Workshop & Interoperability with Discourse*, pp. 135–139, Sofia, Bulgaria.

Tratz, S. C. (2014). Accurate Arabic Script Language/Dialect Classification. Tech. rep., Army Research Laboratory.

Trieschnigg, D., Hiemstra, D., Theune, M., de Jong, F., & Meder, T. (2012). An Exploration of Language Identification Techniques for the Dutch Folktale Database. In *Proceedings of the LREC workshop Adaptation of Language Resources and Tools for Processing Cultural Heritage*, pp. 47–51, Istanbul, Turkey.

Tromp, E., & Pechenizkiy, M. (2011). Graph-Based N-gram Language Identification on Short Texts. In *Proceedings of the 20th Annual Belgian Dutch Conference on Machine Learning (Benelearn 2011)*, pp. 27–34, The Hague, Netherlands.

Ueda, Y., & Nakagawa, S. (1990). Prediction for Phoneme/Syllable/Word-Category and Identification of Language using HMM. In *Proceedings of the 1990 International Conference on Spoken Language Processing, volume 2*, Vol. 2, pp. 1209–1212, Kobe, Japan.

Ullman, E. (2014). Shibboleth - A Multilingual Language Identifier. Master's thesis, Uppsala University, Uppsala.

Unicode (2019). Unicode Character Database. `http://www.unicode.org/ucd/`. Accessed: 2019-07-18.

van der Lee, C., & Bosch, A. v. d. (2017). Exploring Lexical and Syntactic Features for Language Variety Identification. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 190–199, Valencia, Spain.

van Noord, G. (1997). *TextCat*. Software available at `https://www.let.rug.nl/~vannoord/TextCat/`.

van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworths, London, UK.

Vatanen, T., Väyrynen, J. J., & Virpioja, S. (2010). Language Identification of Short Text Segments with N-gram Models. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, pp. 3423–3430, Valletta, Malta.

Vega, V. B., & Bressan, S. (2001a). Continuous-Learning Weighted-Trigram Approach for Indonesian Language Distinction: A Preliminary Study. In *Proceedings of 19th International Conference on Computer Processing of Oriental Languages (ICCPOL 2001)*, Seoul, Korea. Oriental Languages Computer Society.

Vega, V. B., & Bressan, S. (2001b). Indexing the Indonesian Web: Language Identification and Miscellaneous Issues. In *Poster Proceedings of the Tenth International World Wide Web Conference (WWW 10)*, Hong Kong.

Vinosh Babu, J., & Baskaran, S. (2005). Automatic Language Identification Using Multivariate Analysis. In *Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2005)*, pp. 789–792, Mexico City, Mexico.

Vitale, T. (1991). An Algorithm for High Accuracy Name Pronunciation by Parametric Speech Synthesizer. *Computational Linguistics*, *17*(3), 257–276.

Vogel, J., & Tresner-Kirsch, D. (2012). Robust Language Identification in Short, Noisy Texts: Improvements to LIGA. In Atzmueller, M., & Andreas, H. (Eds.), *Proceedings of the 3rd International Workshop on Mining Ubiquitous and Social Environments (MUSE)*, pp. 43–50, Bristol, UK.

Voss, C., Tratz, S., Laoudi, J., & Briesch, D. (2014). Finding Romanized Arabic Dialect in Code-Mixed Tweets. In *Proceedings of the 9th International Conference on Language Resources and Evaluation, ELRA*, pp. 188–199, Reykjavik, Iceland.

Wan, A. (2016). Leveraging Data-Driven Methods in Word-Level Language Identification for a Multilingual Alpine Heritage Corpus. In *Proceedings of the Workshop on Multilingual and Cross-lingual Methods in NLP*, pp. 45–54, San Diego, California.

Wechsler, M., Páraic, S., & Schäuble, P. (1997). Multi-language Text Indexing for Internet Retrieval. In *Proceeding RIAO' 97 Computer-Assisted Information Searching on Internet*, pp. 217–232, Montreal, Canada.

Weiss, D. (2013). *langid-java*. Software available at `https://github.com/carrotsearch/langid-java` (last updated on June 2013).

Williams, J., & Dagli, C. K. (2017). Twitter Language Identification Of Similar Languages And Dialects Without Ground Truth. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 73–83, Valencia, Spain.

Windisch, G., & Csink, L. (2005). Language Identification Using Global Statistics of Natural Languages. In *Proceedings of the 2nd Romanian-Hungarian Joint Symposium on Applied Computational Intelligence (SACI)*, pp. 243–255, Timisoara, Romania.

Wolpert, D. H. (1992). Stacked Generalization. *Neural Networks*, *5*(2), 241–259.

Wray, S. (2018). Classification of Closely Related Sub-dialects of Arabic Using Support-Vector Machines. In *Proceedings of Language Resources and Evaluation (LREC)*, Miyazaki, Japan.

Xafopoulos, A., Kotropoulos, C., Almpanidis, G., & Pitas, I. (2004). Language Identification in Web Documents Using Discrete HMMs. *Pattern Recognition*, *37*(3), 583–594.

Xia, F., Lewis, C., & Lewis, W. D. (2010). The Problems of Language Identification within Hugely Multilingual Data Sets. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, pp. 2790–2797, Valletta, Malta.

Xia, F., Lewis, W. D., & Poon, H. (2009). Language ID in the Context of Harvesting Language Data off the Web. In *+EACL2009*, pp. 870–878, Athens, Greece.

Xia, M. X. (2016). Codeswitching Language Identification Using Subword Information Enriched Word Vectors. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pp. 132–136, Austin, TX, USA. Association for Computational Linguistics.

Xu, F., Wang, M., & Li, M. (2016). Sentence-level Dialects Identification in the Greater China Region. *International Journal on Natural Language Computing (IJNLC)*, *5*(6).

Yaghan, M. A. (2008). "Arabizi" : A Contemporary Style of Arabic Slang. *Design Issues*, *24*(2), 39–52.

Yamaguchi, H., & Tanaka-Ishii, K. (2012). Text Segmentation by Language Using Minimum Description Length. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pp. 969–978, Jeju Island, Korea.

Yang, X., & Liang, W. (2010). An N-Gram-and-Wikipedia joint approach to Natural Language Identification. In *Proceedings of the 4th International Universal Communication Symposium (IUCS 2010)*, pp. 332–339, Beijing, China.

Yeong, Y.-L., & Tan, T.-P. (2010). Language Identification of Code Switching Malay-English Words Using Syllable Structure Information. In *Proceedings of the 2nd International Workshop on Spoken Languages Technologies for Under-resourced Languages (SLTU'10)*, pp. 142–145, Penang, Malaysia.

Yeong, Y.-L., & Tan, T.-P. (2011). Applying Grapheme, Word, and Syllable Information for Language Identification in Code Switching Sentences. In *Proceedings of the International Conference on Asian Language Processing (IALP 2011)*, pp. 111–114, Penang, Malaysia. IEEE.

You, J.-L., Chen, Y.-N., Chu, M., Soong, F. K., & Wang, J.-L. (2008). Identifying Language Origin of Named Entity with Multiple Information Sources. *Audio, Speech, and Language Processing, IEEE Transactions on*, *16*(6), 1077–1086.

Zaidan, O. F., & Callison-Burch, C. (2011). The Arabic Online Commentary Dataset: An Annotated Dataset of Informal Arabic with High Dialectal Content. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pp. 37–41, Portland, Oregon, USA.

Zaidan, O. F., & Callison-Burch, C. (2014). Arabic Dialect Identification. *Computational Linguistics*, *40*(1), 171–202.

Zamora, J. D., Bruzón, A. F., & Bueno, R. O. (2014). Tweets Language Identification Using Feature Weighting. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, pp. 30–34, Girona, Spain.

Zampieri, M. (2013). Using Bag-of-words to Distinguish Similar Languages: How Efficient Are They?. In *Proceedings of the 2013 IEEE 14th International Symposium on Computational Intelligence and Informatics (CINTI)*, pp. 37–41, Budapest, Hungary.

Zampieri, M. (2016). Automatic Language Identification. In *Working with Text: Tools, Techniques and Approaches for Text Mining*, chap. 8, pp. 189–205. Elsevier.

Zampieri, M., & Gebre, B. G. (2012). Automatic Identification of Language Varieties: The Case of Portuguese. In *Proceedings of The 11th Conference on Natural Language Processing (KONVENS 2012)*, pp. 233–237, Vienna, Austria.

Zampieri, M., & Gebre, B. G. (2014). VarClass: An Open-source Language Identification Tool for Language Varieties. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, pp. 3305–3308, Reykjavik, Iceland.

Zampieri, M., Gebre, B. G., Costa, H., & Genabith, J. v. (2015). Comparing Approaches to the Identification of Similar Languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 66–72, Hissar, Bulgaria.

Zampieri, M., Gebre, B. G., & Diwersy, S. (2013). N-gram Language Models and POS Distribution for the Identification of Spanish Varieties. In *Proceedings of la 20ème conférence du Traitement Automatique du Langage Naturel (TALN)*, pp. 580–587, Sables d' Olonne, France.

Zampieri, M., Malmasi, S., Ljubešic, N., Nakov, P., Ali, A., Tiedemann, J., Scherrer, Y., & Aepli, N. (2017). Findings of the VarDial Evaluation Campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 1–15, Valencia, Spain.

Zampieri, M., Malmasi, S., Sulea, O.-M., & Dinu, L. P. (2016). A Computational Approach to the Study of Portuguese Newspapers Published in Macau. In *Proceedings of the Workshop on Natural Language Processing meets Journalism (NLPMJ 2016)*, pp. 47–51, New York City, NY, USA.

Zampieri, M., Tan, L., Ljubešić, N., & Tiedemann, J. (2014). A Report on the DSL Shared Task 2014. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, pp. 58–67, Dublin, Ireland.

Zampieri, M., Tan, L., Ljubešić, N., Tiedemann, J., & Nakov, P. (2015). Overview of the DSL Shared Task 2015. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pp. 1–9, Hissar, Bulgaria.

Zecevic, A., & Vujicic-Stankovic, S. (2013). The Mysterious Letter J. In *Proceedings of the Workshop on Adaptation of Language Resources and Tools for Closely Related Languages and Language Variants*, pp. 40–44, Hissar, Bulgaria.

Zhang, W., Clark, R. A. J., Wang, Y., & Li, W. (2016). Unsupervised Language Identification Based on Latent Dirichlet Allocation. *Computer Speech and Language*, *39*, 47–66.

Zhdanova, A. V. (2002). Automatic Identification of European Languages. In *Revised Papers of the Natural Language Processing and Information Systems - 6th International*

Conference on Applications of Natural Language to Information Systems, (NLDB 2002), pp. 76–84, Stockholm, Sweden.

Zipf, G. K. (1932). *Selected Studies of the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, Massachusetts.

Zirikly, A., Desmet, B., & Diab, M. (2016). The GW/LT3 VarDial 2016 Shared Task System for Dialects and Similar Languages Detection. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 33–41, Osaka, Japan.

Zubiaga, A., San Vicente, I., Gamallo, P., Pichel, J. R., Alegria, I., Aranberri, N., Ezeiza, A., & Fresno, V. (2014). Overview of TweetLID: Tweet Language Identification at SEPLN 2014. In *Proceedings of the Tweet Language Identification Workshop 2014 co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, pp. 1–11, Girona, Spain.

Zubiaga, A., Vicente, I. S., Gamallo, P., Pichel, J. R., Alegria, I., Aranberri, N., Ezeiza, A., & Fresno, V. (2016). TweetLID: A Benchmark for Tweet Language Identification. *Language Resources and Evaluation*, *50*(4), 729–766.