electronics

MDPI

Article

Automatic Multilingual Stopwords Identification from Very Small Corpora

Stefano Ferilli

Department of Computer Science, University of Bari, Via E. Orabona 4, 70125 Bari, Italy; stefano.ferilli@uniba.it;
Tel.: +39-080-544-2293

Abstract: Tools for Natural Language Processing work using linguistic resources, that are language-specific. The complexity of building such resources causes many languages to lack them. So, learning them automatically from sample texts would be a desirable solution. This usually requires huge training corpora, which are not available for many local languages and jargons, lacking a wide literature. This paper focuses on *stopwords*, i.e., terms in a text which do not contribute in conveying its topic or content. It provides two main, inter-related and complementary, methodological contributions: (i) it proposes a novel approach based on term and document frequency to rank candidate stopwords, that works also on very small corpora (even single documents); and (ii) it proposes an automatic cutoff strategy to select the best candidates in the ranking, thus addressing one of the most critical problems in the stopword identification practice. Nice features of these approaches are that (i) they are generic and applicable to different languages, (ii) they are fully automatic, and (iii) they do not require any previous linguistic knowledge. Extensive experiments show that both are extremely effective and reliable. The former outperforms all comparable approaches in the state-of-the-art, both in terms of performance (Precision stays at 100% or nearly so for a large portion of the top-ranked candidate stopwords, while Recall is quite close to the maximum reachable in theory.) and in smooth behavior (Precision is monotonically decreasing, and Recall is monotonically increasing, allowing the experimenter to choose the preferred balance.). The latter is more flexible than existing solutions in the literature, requiring just one parameter intuitively related to the balance between Precision and Recall one wishes to obtain.

Keywords: natural language processing; machine learning; stopword identification

check for updates

Citation: Ferilli, S. Automatic Multilingual Stopwords Identification from Very Small Corpora. *Electronics* **2021**, *10*, 2169. https://doi.org/10.3390/electronics10172169

Academic Editors: Juan M. Corchado, Stefanos Kollias and Javid Taheri

Received: 30 July 2021
Accepted: 2 September 2021
Published: 5 September 2021

1. Introduction and Motivation

Natural Language Processing (NLP) is the branch of Artificial Intelligence concerned with automatic processing of text written in natural languages, so as to improve human fruition of its contents. Several NLP tasks are associated to the different, and increasingly complex, levels at which natural languages may be studied and analyzed (morphology, lexicon, grammar, syntax, semantics, and pragmatics). NLP tools typically rely on linguistic resources (lists of words or suffixes, etc.). Since each language is different from the others, specific resources are needed for each language. Manual development of the resources by linguistic experts is time-consuming (it takes much study and refinement), costly (expert time is valuable), and potentially erratic (as all human activities).

While several examples exist for the various NLP tasks, here, we will just provide one that is representative of all of the above issues. A very famous linguistic resource, widely exploited for many tasks, is WordNet, a semantic network/lexical taxonomy (some consider it an ontology) that organizes concepts and English terms according to several syntactic or semantic relationships [1]. Being manually developed, it is not free of bugs. The problem of cost is evident in the current Webpage of the project (https://wordnet.princeton.edu/, accessed on 31 August 2021), which states: "Due to funding and staffing issues, [...] there are currently no plans for future WordNet releases." Moreover, the effort had to be duplicated for many other languages, with mixed results. The EuroWordNet project [2] considered Dutch, Italian, Spanish, German, French, Czech, and Estonian. Its homepage (https://archive.illc.uva.nl/EuroWordNet/, accessed on 30 July 2021) reports on efforts for Swedish, Norway, Danish, Greek, Portuguese, Basque, Catalan, Romanian, Lithuan, Russian, Bulgarian, and Slovenic. Another work in the same direction is MultiWordNet [3], that aligns the Italian WordNet with Princeton WordNet and also provides access to the Spanish, Portuguese, Hebrew, Romanian, and Latin WordNets, developed by other research groups. It demonstrates another problem in the building and development of language-specific linguistic resources: the additional work needed to keep them aligned when new versions are available (it is aligned with WordNet v1.6 but subsequent WordNet versions are not backward compatible).

So, it would be desirable to automatically learn the resources from texts in a given language, but finding effective strategies is not trivial, and often relies on statistics that must be drawn from very large amounts of text. In fact, many good resources have been developed for English; some, not always very reliable, resources are available for a few major languages, but nearly nothing exists for the vast majority of non-widespread languages, dialects, and jargons. This prevents application of NLP to the latter, which tampers cultural diversity and might even cause extinction of languages in the long run, with a consequent huge cultural loss.

An NLP task working at the lexical level is Stopword Identification, where “Stopwords are terms that occur most frequently in a document and contain very little information that is usually not necessary” [4]. Ref. [5] defines stopwords by contrast to most significant words (‘keywords’) as follows: “Since significance is difficult to predict, it is more practical to isolate it by rejecting all obviously non-significant or ‘common’ words, with the risk of admitting certain words of questionable status. Such words may subsequently be eliminated or tolerated as so much ‘noise’. ” The historical and main application that needs stopwords is Information Retrieval (IR). It aims at indexing a corpus of texts so as to quickly and accurately select those that may best satisfy the information needs of users, usually expressed by queries consisting of sets of terms, e.g., effective IR may foster cultural diversity, and keep alive languages, by making documents in those languages easily retrievable by end-users. IR mostly works at the lexical level, and carries out Stopword Removal to reduce the number of indexed terms. This increases speed, reduces storage requirements, and improves accuracy (by focusing the index on meaningful terms only). Today, a significant trend for IR is based on techniques that avoid text pre-processing and, thus, stopword removal, as well. However, these techniques require very large amounts of data to be applicable, which would not feasible under the small-corpora setting. Additionally, many other applications still exist that rely on Stopword Identification, e.g., it is still relevant or necessary for linguistic studies, or for supporting applications, such as Diachronic Analysis (concerning lexical and semantic changes in language along time) [6] and Sentiment Analysis (see, e.g., References [7,8]).

The linguistic resource used for Stopword Removal is known as ‘stopword list’. It consists of the list of terms to be found and removed from the texts. Quoting, again, Reference [5], “A list of nonsignificant words would include articles, conjunctions, prepositions, auxiliary verbs, certain adjectives”, known as ‘function words’. This approach requires grammatical knowledge of the language, which might not be available. In addition, additional specific terms might be insignificant in domain-specific contexts (e.g., “words, such as “report”, “analysis”, “theory”, and the like” are considered as irrelevant in the domain indexing of technical literature [5]). So, each specific domain has its own stopwords, but only generic stopword lists are usually developed. Even worse, an analysis revealed that well-known and widely used stopword lists are not very accurate, which clearly affects their effectiveness [9].

Motivated by all the considerations above, this paper proposes an integrated approach to automatically learn stopword lists, made up of two components:

- a simple yet effective frequency-based approach to rank candidate stopwords, and
- a geometric strategy to determine the cutpoint in this ranking.

Importantly and interestingly, our approach can work under the following constraints:

1. using just plain texts (i.e., no previous linguistic knowledge): so as to deal with languages for which no formal grammar is available;
2. being language-independent (not tailored to a specific language): so as to provide its benefits to a whole range of languages;
3. working on *very small* corpora (in extreme cases, even a single text): so as to deal with languages having limited spread or literature (to the best of our knowledge, this setting is original; all previous works in the literature assumed huge amounts of data to be available, which is not always true in practice); and
4. being fully automatic: so as to avoid the shortcomings of using linguistic experts.

We ran experiments on several languages of different complexity, and even on mixed languages, proving the effectiveness of our proposal and obtaining interesting insight about the problem in general and how to practically apply our approach.

In the following, after discussing related works, we will describe our approaches, our experimental setting and the datasets used. Then, we will discuss our experimental results on stopword extraction from very small corpora and single texts in different languages, before concluding the paper.

2. Related Work

Since stopword removal is very relevant for IR, proposals for automatic stopword extraction were often evaluated indirectly through the performance of IR based on the extracted stopwords (e.g., Reference [10]). In this paper, we are mainly interested in the linguistics perspective; thus, we will evaluate the extracted stopword lists based on their contents, rather than on their performance on other tasks.

Some works learn the stopword lists based on external aids, such as labeled texts or extant language-specific tools/resources, e.g., Ref. [10] uses a Vector Space Model, but previously applies stemming. Ref. [11] also applies Porter's algorithm for stemming the text and adopts a supervised learning approach. In addition, Reference [12] works on labeled corpora. Ref. [13] focuses on the task of optimizing an existing stopword list, with an approach based on the entropy of words. Ref. [14] also adopts a supervised approach. Ref. [15] exploits Part-of-Speech information. These approaches cannot be directly compared to our proposal, in which we purposely start from plain text and avoid any kind of aid or pre-processing.

Refs. [16,17] proposed two approaches purely based on frequency. Both are language-specific (English and French, respectively), and both involve manual adjustment of the list of stopwords extracted automatically. The former was tested on a corpus of broad literature including more than 1 million words, while the latter was applied to two corpora made up of small texts, including more than 4 and more than 6 million words, respectively. In addition, Reference [12] proposes "automatic generation of domain-specific stopwords from a *large labeled* corpus". We aim at learning stopwords from much less data.

Let us now introduce some notation. We will denote by $\mathcal{C}$ the training corpus, in a given language, for learning a stopword list for that language, by $n = |\mathcal{C}|$ the number of texts in $\mathcal{C}$, and by $V = \{t_1, \ldots, t_m\}$ the *vocabulary* of $\mathcal{C}$, i.e., the set of distinct terms used in $\mathcal{C}$. For each term $t_i \in V$, o_i denotes the number of its occurrences in $\mathcal{C}$, n_i the number of texts in which it occurs, and o_i^c the number of its occurrences in text $c \in \mathcal{C}$. So, $o = \sum_i o_i$ denotes the total number of *tokens* (i.e., occurrences of terms) in $\mathcal{C}$. In the following, we will consider $\mathcal{C}$ as fixed and, thus, will ignore it in the notation.

Many techniques proposed in the literature work by ranking all terms in the collection according to their degree of 'stopwordness', based on Zipf's law (the relation $F(r) = \frac{C}{r^\alpha}$ with $\alpha \approx 1, C \approx 0.1$ describes very precisely the distribution of frequency of terms rank). Then, they select the top terms in the ranking according to Algorithm 1. Some such techniques are supervised, e.g., Reference [11]: Information Gain, χ^2 Statistic, Odds Ratio, and F-measure Feature Ranking. We assume no information is available except the plain text(s); thus, we cannot exploit such approaches. Instead, we turn to unsupervised approaches in the following. Different functions $f(t)$ used by unsupervised approaches in the literature are:

- Term frequency (TF): the number of times a term occurs in the corpus:

$$f(t_i) = \mathrm{tf}(t_i) = o_i.$$

- Normalized Term Frequency (NTF): TF normalized with respect to the total number of tokens in the corpus:

$$f(t_i) = \mathrm{ntf}(t_i) = -\log(\frac{o_i}{o}).$$

- Inverse Document Frequency (IDF) [18]: based on the number of texts in the corpus in which the term occurs (assuming that the more texts use a term, the less informative it is):

$$f(t_i) = \mathrm{idf}(t_i) = \log(\frac{n}{n_i})$$

- Normalized IDF (NIDF): IDF normalized with respect to the number of texts that do not contain the term $(n - n_i)$, with a 0.5 adjustment to mitigate extreme values [19]:

$$f(t_i) = \mathrm{nidf}(t_i) = \log(\frac{(n - n_i) + 0.5}{n_i + 0.5}).$$

- Entropy (H): based on the distribution of a certain term over the documents collection, i.e., on how (un)evenly distributed it is in the corpus:

$$f(t_i) = \mathrm{h}(t_i|\mathcal{C}) = -\sum_{c \in \mathcal{C}} P(c|t_i) \log P(c|t_i),$$

where $P(c|t_i) = o_i^c / o_i$ (we recall that o_i^c is the number of occurrences of term t_i in document c). The terms having higher entropy contain less information about the documents where they appear, than terms with lower entropy. The maximum entropy value for a given collection of documents is $\log |\mathcal{C}|$, obtained for an even distribution.

A different, and more complex, approach is Term-based Random Sampling (TRS) [20]. It randomly selects n terms, and, for each, produces a set of candidate stopwords as follows: it samples all the texts containing the term and assesses the relevance of each term t in the sample using the KL divergence measure [21]:

$$d_x(t) = P_x(t) \cdot \log_2 \frac{P_x(t)}{P(t)},$$

to compare its distribution within the sample and in the whole corpus, where:

- $P_x(t)$ is the normalized frequency of t within the sampled texts;
- $P(t) = o_i / o$ is the normalized frequency of t in the whole corpus.

Then, each set of candidate stopwords is shrunk to m items, and the l least informative candidates overall are returned as stopwords. So, TRS requires 3 input parameters (n, m, l). Note that terms rarely occurring in the collection are likely to yield a small set of terms because few texts contain them. So, the samples obtained by selecting n should improve the estimation of the distribution and relevance of terms. Due to its random nature, the behavior of TRS is very variable and hard to capture.

Ref. [20] compared the performance of TRS in supporting IR to TF, NTF, IDF, and NIDF, reporting NIDF to be the best technique. However, Ref. [11] points out the deficiencies of the DF-based approach in general. We further note that, for collections consisting of very few texts, they cannot leverage enough variability and are inapplicable in the case of just one document. Being based on the distribution of terms across texts, H alsosuffers from the same problems and limitations. On the other hand, using TF, Ref. [9] uncovered some flaws in standard stopword lists in the literature. Inspired by References [9,22], Ref. [23] has shown that TF dramatically outperforms both NIDF and TRS on small corpora, as well as extensively discussed the behavior of these techniques on different types of texts.

Given a training corpus, these approaches will typically yield different stopword lists.

As witnessed by the most recent survey paper available on Stopword Removal, Reference [4], the literature after Reference [20] mainly focused on specific and peculiar languages, especially those using non-Latin script. A list of such works (often published in National conferences or journals) includes Arabic [24–26], Chinese [27,28], Persian [15], Sanskrit [29], Gujarati [30], Punjabi [31], Hindi [32–34], Bengali [35], Sinhala [36], and Tamil [37]. Here, we aim at devising an approach that can be applied to different languages; thus, we will not discuss these works in the following, nor can we compare our proposal to these works, which use very tailored approaches.

Since most approaches to automatic stopword identification work according to Algorithm 1, a relevant problem is how to determine the cutoff threshold θ. This is not trivial, due to the typical shape of the $f(t)$ plot providing little hints to determine the cutpoint. While not reporting the value used to obtain the best performance, Ref. [20] proposed to determine θ using the largest frequency difference between adjacent terms in the ranking: if it happens between frequencies $f(r)$ and $f(r+1)$, they take $\theta = f(r)$. However, this might be misleading, since the maximum difference typically happens quite early in the ranking, and might cut away too many stopwords. Ref. [22] used the average-based threshold $\theta = \frac{\alpha}{n}\sum_{i=1}^{n} t_i$, and experimentally found that $\alpha = 1.05$ yields good results. Again, this solution does not consider the whole shape of the frequency ranking. In other applications facing the same problem, the derivative is used to cut the list, where the plot becomes (almost) flat. This is misleading, too, because the plot is irregular, and (especially in short texts) it often becomes nearly flat for a short time, with no strict relationship to the stopwords.

Algorithm 1 Ranking-based stopword identification algorithm.

```
Require: vocabulary $V$ for $\mathcal{C}$
Require: threshold $\theta$
Ensure: S /* set of identified stopwords */
  $S \leftarrow \emptyset$
  for all $t \in V$ do
    if $f(t) \geq \theta$ then
      $S \leftarrow S \cup \{t\}$
    end if
  end for
  return S
```

3. Proposed Approach

Based on its very definition, 'stopwordness' of a term is proportional to its frequency of occurrence. The most straightforward interpretation of this is considering its term frequency. In addition, indeed, the TF approach, albeit simple, demonstrated great potential, especially in the case of very few training documents [23]. In another interpretation, the presence of a term in many documents may also be an indication of its being irrelevant to distinguish them. Strangely enough, however, in the literature, the Document Frequency (DF) of a term, i.e., the number of documents in which it appears, has always been used in its inverse form (IDF), possibly normalized (NIDF). Perhaps the inspiration for this came from the usual weighting schemes adopted for the Vector Space Model in IR (e.g., TF*IDF), where the more the spread of a term in the corpus, the less its relevance to a single document. However, using DF in the denominator decreases the weight of a term appearing in many documents, while, in our perspective, a term occurring in many documents should increase the degree of stopwordness for that term. In fact, the classical Vector Space Model expresses the relevance of terms to specific documents, while, here, we are interested in its relevance in the whole collection. In addition, it is unclear why TF, which is the parameter most obviously associated to stopwordness, was ignored by Reference [20] when using IDF.

Based on these considerations, we purport that DF and TF can support each other in stopword identification and propose to combine them so that DF is directly, rather than inversely, proportional to 'stopwordness'. So, each term will be associated with a value equal to the product of TF and DF. We call this extension Term-Document Frequency (TDF), defined by the following function to be used in Algorithm 1:

$$f(t_i) = \mathrm{tdf}(t_i) = o_i \cdot n_i.$$

To the best of our knowledge, no one investigated this approach so far.

As to the choice of the cutpoint for the list of candidate stopwords, we propose a geometric strategy, formally described in Algorithm 2 and graphically shown in Figure 1, where the x axis reports the term ranking positions for terms ordered by decreasing stopwordness according to the f function, and the y axis reports the stopwordness values. First, all *different* values of the stopwordness function $f(t)$ for the terms in the corpus are considered, ordered by decreasing value, and plotted on a Cartesian space where the x-axis is associated to the set of different frequencies and the y-axis reports the actual frequencies (plot in Figure 1). Considering only different values avoids plateaus, yielding a monotonically decreasing plot and a more compact x-axis, especially on its trailing part. The resulting diagram has an irregular hyperbole-like shape, on which the cut point is determined geometrically as follows. Consider a line $y = ax + b$ of given (decreasing) slope a, and height b on the origin of the Cartesian space such that it is tangent to the plot. Then, the y coordinate of the tangency point will be our cutpoint frequency. So, our procedure takes the slope parameter a as the only input. Acting on a, one may obtain a stricter or looser selection: the more the slope, the earlier the cutpoint; the less the slope, the later the cutpoint. Of course, being the plot on the positive quadrant, we will consider negative slopes. Since the boundaries of the plot depend on the number of words in the vocabulary (x-axis) and on the maximum word frequency in the corpus (y-axis), the same line slope will have different effects depending on such boundaries. To reduce the effect of this issue, we propose to normalize the axes ranges, so that the plot becomes square. This has a nice side-effect on the understandability of the slope setting. Indeed, parameter $\overline{a} = -1$ corresponds to a $-45°$ slope, which should identify the cutpoint in which the plot slope changes from vertical to horizontal. So, values $a \in]-1, 0]$ will select the cutpoint on the right-hand side of the elbow, while $a \in]-\infty, -1[$ will select the cutpoint on the left-hand side of the elbow.

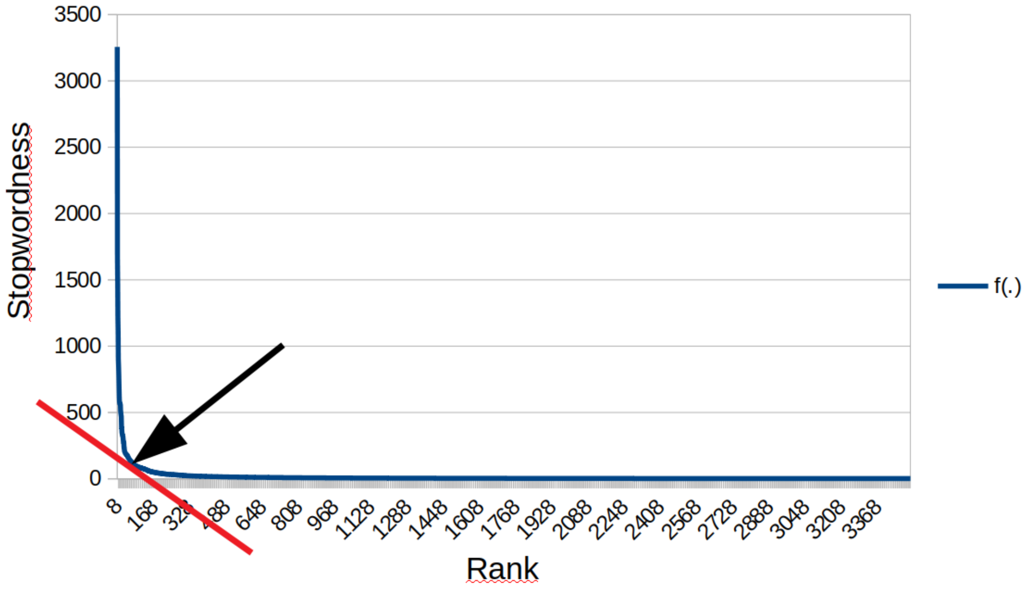

Figure 1. Geometric cutpoint assessment.

Algorithm 2 Cutpoint assessment.

```
Require: $V$ : vocabulary for $\mathcal{C}$
Require: $a$: slope of the line determining the cutpoint
Ensure: $S$ /* set of identified stopwords */
  $D \leftarrow \{f(t)|t \in V\}$ /* all different stopwordness values */
  $D' \leftarrow \{\frac{d \cdot |D|}{\max(D)}|d \in D\}$ /* normalization to make the plot square */
  $L \leftarrow < l_1, \ldots, l_{|S|} >$ list of values in $D$ in decreasing order
  $L' \leftarrow < l'_1, \ldots, l'_{|S|} >$ list of (normalized) values in $D'$ in decreasing order
  $i \leftarrow 1$ /* index of the cutpoint value */
  $continue \leftarrow$ True
  while $continue$ do
    $b \leftarrow l'_i - a \cdot i$
    $j \leftarrow i + 1$
    while $j \leq |D| \wedge l'_j > aj + b$ /* $j$-th normalized bar above the line */ do
      $j \leftarrow j + 1$
    end while
    if $j \leq |D|$ then
      $i \leftarrow j$ /* found new candidate cutpoint value */
    else
      $continue \leftarrow$ False /* all values processed */
    end if
  end while
  $S \leftarrow \emptyset$
  for all $t \in V$ do
    if $f(t) \geq l_i$ then
      $S \leftarrow S \cup \{t\}$
    end if
  end for
  return S
```

We may consider the frequency plot as composed by the top of the bars of a bar diagram where the bar at the i-th position on the x-axis represents the i-th different frequency (in decreasing order). In this representation, the tangent line we are looking for will have the following properties:

- passing from a point $(\bar{i}, f(t_{\bar{i}}))$ for some term (i.e., candidate stopword) $t_{\bar{i}}$;
- being below all bar tops in the diagram: $\forall i, i \neq \bar{i} : f(t_i) \geq f(t_{\bar{i}})$.

It can be found by analyzing, in turn, each i, computing the corresponding b ($= f(t_i) - ai$), and checking that the bar top of all other i's is above the line, i.e., $f(t_i) \geq ai + b$. Operationally, we proceed for increasing i's, starting from $i = 1$. For each i under consideration we scan the subsequent bar tops, and skip them while they are above the line. If we reach the end of the plot, then the current i provides the cutpoint frequency; otherwise, as soon as we find a bar whose top is below the line, the corresponding position becomes our current candidate i, and we go back to the skipping step. In practice, instead of normalizing the axes, we report the normalization on the slope, so as to avoid recomputing all bar values, and also still having all integer values. Consider slope $\bar{a}$ for the case of equal x and y ranges. With different x and y ranges, say $[0, \max_x]$ and $[0, \max_y]$, respectively, the unitary increase of x corresponds to a $\max_y$ / $\max_x$ increase of y; thus, the normalized slope to be used in practice is $\bar{a} \cdot \max_y$ / $\max_x$.

The procedure is shown in Figure 2 for a sample bar diagram, where the steps are denoted by numbers in circles. The axes in Figures 2 and 3 were not labeled because they describe our cutpoint assessment approach in general, for any monotonically decreasing histogram, independent of its interpretation —stopwordness ranking or other (in this work, they are to be interpreted as in Figure 1). The bars were enlarged for the sake of readability, so that their points become squares; let us consider the centroid of each square as point represented by the square. Circled numbers below the diagrams denote the various steps of the procedure, and, for each step, an arrow shows the candidate cutpoint. Bars whose top point falls below the line are filled with gray in the picture. (1) We start with the first bar, and draw the line passing from its top point (centroid of the top square); we start scanning the subsequent bar tops and see that the second bar top is already below the line. (2) So, the second bar top becomes our new candidate; the line passing from it is drawn, and scanning of subsequent bar tops starts; again, the next (third) bar top is below the line. (3) The line passing from the third bar top is drawn, and scanning starts: the next (fourth) bar top is above the line, so it is skipped, while the fifth bar top is below the line. So, step (4) is not carried out and our new candidate becomes the fifth bar top. (5) The line passing from the fifth bar top is drawn, and scanning starts: all subsequent (sixth and seventh) bars are above the line, and then the bar diagram ends. So, the selected cutpoint corresponds to the height of the fifth bar.

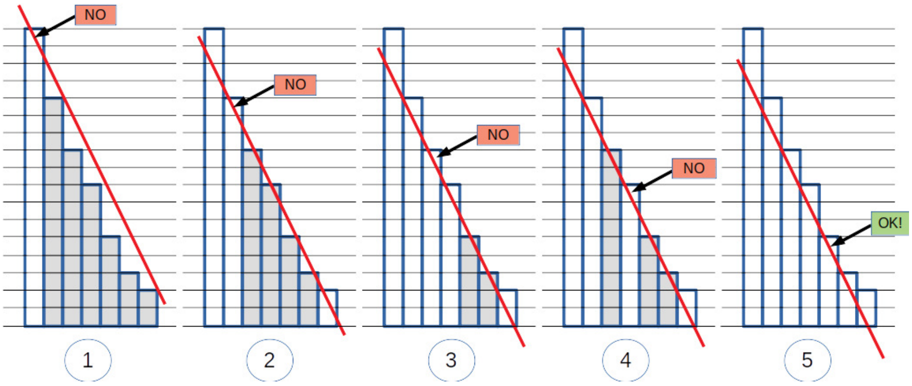

Figure 2. Steps of the geometric cutpoint assessment procedure.

Note that the lines in the various attempts all have the same slope, provided as an input parameter. Of course, changing the slope might return different results. Figure 3 shows the tangent lines for 3 different input slopes, along with the corresponding cutpoints (pointed by the arrows). In addition, it may happen that the line is tangent to the plot in more than one point (e.g., in the rightmost case in Figure 3, the tangent passes from the top of the sixth and seventh bar). In such a case, different strategies can be applied to determine which of these points should be selected as the cutpoint. Our strategy returns the earliest bar for which the line is below all the bars, which yields the strictest strategy, returning less stopwords. Other possible selection strategies are: the latest (loosest strategy, returning most stopwords), the middle one, etc.

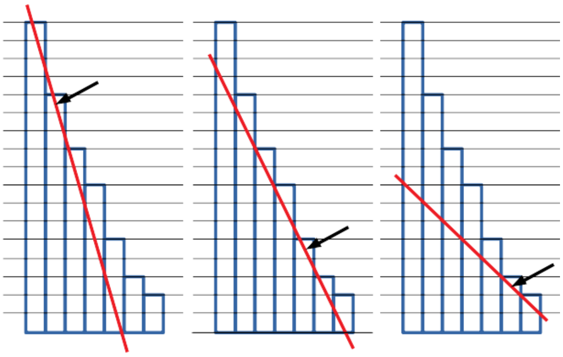

Figure 3. Cutpoint assessment for different slopes.

4. Experimental Setting

To test our TDF and cutoff threshold assessment approaches, we devised an experimental setting compliant with the constraints stated in the Introduction:

1. We consider plain texts, each associated to one language. Words or phrases from other languages, if any, will act as noise.
2. We evaluated our proposed methods on 3 languages:
    - English, as the main language for which NLP solutions have been developed in the literature;
    - Italian, as an important language with a much more complex morphology than English (its much richer inflection might affect frequency-based stopword identification—thus, if good results are obtained on Italian, one may expect to obtain good results also on many other languages), for which NLP solutions are also available;
    - Squinzanese, a Southern Italy dialect already investigated in Reference [22], as an example of a dialect for which no linguistic resources are available, and few texts are available to learn them.

    We also tested our approach in a multilingual setting, on a corpus obtained by merging the English and Italian corpora.
3. We selected *very* small corpora (including up to 18 texts) for each language. This will make learning more difficult than on a large number of texts, where the frequency of real stopwords should easily dominate that of the other words.
4. Our approach is fully automatic.

For each language, we used narrative texts of different length from traditional literature. For English and Italian, we also selected additional texts in more specific styles (technical, poetry, drama —these texts were aimed at stressing our approach so as to analyze its behavior under different conditions, inspired by Reference [23], while mainstream literature typically focused on texts in the same style. Indeed, some are skeptical about the use of mixed styles, especially when the number of texts for each style is so small. For the dialect they were not available. Narrative texts are from the XIX or beginning of the XX century; poetry/drama texts are from the Middle Ages; technical texts are from the late XX century. Many selected texts contain typos because they were extracted using Optical Character Recognition on scanned images of paper documents. This further noise in the data makes our experimental setting more similar to real-world cases.

Due to the different nature of the texts, differently from the dialect, we planned a strategy for progressively incrementing the dataset in the experiment.

Tables 1–3 report the single texts and the aggregates that were used in the experiments for each language, each associated with an identifier (*ID*), to be used for referencing it in the rest of the paper, and with some statistics. Specifically, we reported their length (in number of words as counted by a text editor), the size of their vocabulary $|V|$ (i.e., the number of different words a *word* is defined here as a sequence of alphabetic characters only, preceded and followed by non-alphabetic characters—, as computed by our code), the number of stopwords #*s* they include from the ground truth, and the recall R_{max} corresponding to such stopwords (i.e., the maximum recall that any stopword identification technique may reach on such documents). The number of words for each text or group of texts in the training corpus is relevant for relating them to performance.

English texts (see Table 1) are mostly novels, plus the works of Shakespeare for poetry/drama, and the DOS manual as a technical text. The novels were collected into 2 groups: 'Dumas', including all serial stories by Alexandre Dumas, Père; and 'Novels' for the rest. Another group ('Tech') included the other texts (in more 'technical' style): the complete works by Shakespeare and the DOS manual. For Italian (see Table 2), the selected texts are a subset of those in Reference [23]. Again, they include mostly narrative texts. Italian texts are generally shorter than English ones, but their vocabulary is generally larger, except for HeG that uses less than 900 distinct words. Again, the narrative texts were collected into two groups: 'PPI' includes the 5 volumes of 'Passeggiate per l'Italia', a report of travels around Italy by a foreign visitor from the XIX century; 'Novels' includes novels and collections of stories from classical Italian literature. Again, another group ('Tech') included the other texts (in more 'technical' style): Dante's poem 'La Divina Commedia' and the collection of civil norms in the Italian law. Squinzanese texts (see Table 3) were taken from a tale book [38] (one of the few available in this language). They are generally much shorter, and with a much smaller vocabulary, than English and Italian ones, which makes this dataset particularly challenging. Since all Squinzanese texts are in the same style, we just collected them in 3 groups consisting of 6 tales each, in the order in which they appear in the book table of contents.

Table 1. English corpus.

ID	Text	Words	$\|V\|$	#s	R_{max}
DJaMH	Dr Jekyll and Mr Hyde	28,820	4287	148	0.85
CC	Captains Corageous	55,943	7024	162	0.93
F	Frankenstein	78,213	7268	153	0.88
TBA	The Black Arrow	82,881	7381	148	0.85
ACYiKAC	A Connecticut Yankee in King Arthur's Court	121,985	10,298	169	0.97
M-D	Moby-Dick	213,788	17,053	164	0.94
TCoMC	The Count of Monte Cristo	466,609	15,935	162	0.93
TTM	The Three Musketeers	233,250	10,529	157	0.90
TYA	Twenty Years After	245,899	11,133	164	0.94
TVdB	The Vicomte de Bragelonne	193,555	10,571	155	0.89
LdlV	Louise de la Valliere	171,636	9320	155	0.89
TYL	Ten Years Later	191,000	9917	157	0.90
TMitIM	The Man in the Iron Mask	177,654	10569	158	0.91
Scw	Shakespeare Complete Works	962,009	25,764	160	0.92
D33	DOS 3.3 manual	108,495	3356	132	0.76
Novels	{DJaMH,CC,F,TBA,ACYiKAC,M-D}	581,630	25,739	172	0.99
Dumas	{TCoMC,TTM,TYA,TVdB,LdlV,TYL,TMitIM}	1,679,603	25,464	165	0.95
Tech	{Scw,D33}	1,070,504	27,350	162	0.93
noTech	Novels ∪ Dumas	2,261,233	35,756	174	1.00
All	noTech ∪ Tech	3,331,737	47,382	174	1.00

Table 2. Italian corpus.

ID	Text	Words	$\lvert V \rvert$	#s	R_{max}
PPI1	Passeggiate per l'Italia 1	71,467	11,995	162	0.58
PPI2	Passeggiate per l'Italia 2	86,818	14,710	170	0.61
PPI3	Passeggiate per l'Italia 3	75,871	12,721	170	0.61
PPI4	Passeggiate per l'Italia 4	75,618	12,183	165	0.59
PPI5	Passeggiate per l'Italia 5	46,655	10,470	162	0.58
L'E	L'Esclusa	55,846	8919	167	0.60
IPS	I Promessi Sposi	220,174	19,658	226	0.81
TlN	Tutte le Novelle	264,703	21,641	229	0.82
HeG	Hansel e Gretel	2485	890	67	0.24
LDC	La Divina Commedia	97,714	12,796	153	0.55
CCI	Codice Civile Italiano	228,251	8659	128	0.46
PPI	{PPI1,PPI2,PPI3,PPI4,PPI5}	288,379	30,855	215	0.77
Novels	{L'E,IPS,TlN,HeG}	487,362	34,677	251	0.90
Tech	{LDC,CCI}	325,965	19,900	190	0.68
noTech	PPI ∪ Novels	775,741	51,062	257	0.92
All	noTech ∪ Tech	1,101,706	60,427	262	0.94

Table 3. Squinzanese corpus.

ID	Text	Words	$\lvert V \rvert$	#s	R_{max}
LFF	Lu Fushi-Fushèi	2125	599	65	0.34
MFF	Mesciu Frangiscu firraru	5126	1133	83	0.43
LMS	La Maria scema	1209	405	40	0.21
LNN	Lu Ndlì-ndlì	4129	939	83	0.43
LR	Lu Ranarieddhru	3272	751	63	0.33
LN	La Nannorca	1056	342	42	0.22
LCS	Lu Cumpare Scaravashu	1844	514	60	0.31
LPZ	La Pecura Zzoppa	1088	331	44	0.23
LJ	Lu Jaddhru	1601	497	58	0.30
BclNeRclS	Bianca comu 'lla Nive e Russa comu 'llu Sangu	911	710	71	0.37
ABA	Angila Bell'Angila	1040	396	38	0.20
MS	Maria Sapiente	3963	850	73	0.38
RF	Rre Fiore	7980	1439	94	0.49
II	Isabbella Isabbellina	3175	752	73	0.38
RS	Rre Sarpente	4247	981	88	0.46
LT	Lu Thriticinu	3579	780	75	0.39
LV	Lu Valanieddhru	1064	353	40	0.21
LT2	Lu Tiaulu	1805	600	60	0.31
1–6	{LFF,MFF,LMS,LNN,LR,LN}	16,917	2393	111	0.58
7–12	{LCS,LPZ,LJ,BclNeRclS,ABA,MS}	10,447	1918	115	0.60
13–18	{RF,II,RS,LT,LV,LT2}	21,850	2731	125	0.65
1–12	1–6 ∪ 7–12	27,364	3298	127	0.66
All	1–12 ∪ 13–18	49,214	4547	138	0.72

As a baseline for performance evaluation, and to get an idea of its performance on the extreme case of just one training document, the proposed approach was applied separately to each single text. Note that, applied to one text, the TDF approach boils down to TF, since $DF = 1$ for all terms. Then, the approach was applied to increasingly larger sets of texts, selected so as to investigate the approach behavior on different kinds of texts (i.e., texts with homogeneous or different styles). First, it was applied to the groups specified above, for investigating its approach on texts of homogeneous style. Then, it was applied to further aggregations: all narrative texts for English and Italian, and the first two groups for Squinzanese. Finally, it was applied to the whole corpus for each language.

Concerning the automatic cutpoint identification for the candidate stopwords, we will test the $-a$ parameter in the $[0,1]$ range, denoting slopes smaller than or equal to 45°. The underlying rationale is that, for greater slopes, there is still a strong decay in word frequency, while we expect non-stopwords to have a more even frequency distribution. Specifically, we will test 3 values: $a = -1$, as the elbow point that distinguishes more quickly varying frequencies from more smoothly varying ones; $a = -0.5$, as the middle point in the interval, and $a = -0.25$, as in between the latter and the horizontal line ($a = 0$, which would take all terms as candidate stopwords).

Many works in the literature, specifically focusing on the IR task, indirectly evaluated the performance of their stopword identification approaches based on the performance of the subsequent IR applications. Since we do not focus exclusively on IR, and we tackle the case of very few texts, we will adopt a content-based evaluation approach, more based on linguistics, and compare the extracted stopwords to those in golden standard stopword lists. As the golden standard for English and Italian, we will use the stopword lists provided by Snowball (https://snowballstem.org/, accessed on 4 September 2021), which are well-known and currently exploited by many NLP systems. The English list includes 174 stopwords, while the Italian list consists of 279 stopwords. Since no golden standard was available for Squinzanese, we used a list obtained by translating the stopwords in the golden standard for Italian, resulting in 192 stopwords. Performance will be evaluated in terms of number of returned candidate ($\#c$) and actual ($\#s$) stopwords, precision (P), recall (R), and sum of precision and recall ($P + R$) with respect to the golden standard. Precision is important to prevent removal of informative words when pre-processing the text. Recall, i.e., how many stopwords from the golden standard are retrieved, allows us to understand whether the results of the automatic technique are comparable to those of human experts. However, we put more emphasis on precision because the very small corpora might not include some stopwords in the golden standard. To have a single number expressing a balance in performance between P and R, we use their sum, instead of the traditional F-measure. This is because F-measure rewards more cases in which precision and recall are close, while, in our case, they are always very imbalanced and, so, would yield very low values for F-measure.

It is worth noting that the reported results can be actually considered a lower bound on performance, since the golden standard is known to miss many stopwords (e.g., in Italian preposition 'fra', an alternate form of 'tra', which is in the list; in English, the archaic form 'thou' for 'you'), especially most truncated forms in Italian (LDC is a poem in archaic Italian from the 1300s, so the most frequent terms are often real stopwords, but truncated for poetry; these truncated forms are missing in the golden standard, but are actually very common also in everyday language, so, this is not an issue with the text, rather it further confirms the incompleteness of the golden standard noted in Reference [9]). Indeed, Reference [9] showed that, considering some missing stopwords, precision of the first 100 candidate stopwords (P@100) rises from 0.72 to 0.94 on the entire dataset, and even more for some texts (see Table 4). Worth noting are the cases LDC and HeG: the former has the best increase in precision, from 0.53 to 0.92, becoming the most effective text in the corpus; in the latter precision also increases by 0.18, up to 0.70, in spite of its being a very short text.

Table 4. Comparison of P@100 for Snowball-based and manual evaluation on Italian texts (from Reference [9]).

Text(s)	LDC	CCI	L'E	IPS	TLN	PPI1	PPI2	PPI3	PPI4	PPI5	PPI	HeG	N-T	All
Snowball	0.53	0.53	0.62	0.65	0.62	0.73	0.71	0.68	0.71	0.66	0.72	0.52	0.69	0.72
Manual	0.96	0.70	0.86	0.90	0.93	0.90	0.87	0.85	0.88	0.86	0.92	0.70	0.89	0.94

5. Experimental Results

This section reports extensive experimentation showing the effectiveness of our proposals for stopword identification (both the ranking function and the cutoff assessment), on different languages and even on mixed languages.

5.1. Comparison to State of the Art

The obvious competitor for the method we propose is TF for two reasons:

1. it proved to be the most effective approach in the state-of-the-art under the constraints we set in our research (small corpora, no manual labeling, no use of external tools and resources) [23]; indeed, experiments on very small corpora in Reference [23] show that TF dramatically outperformed the best similar state-of-the-art approaches proposed in Reference [20]: precision was basically 0 up to the first 100 candidate stopwords identified by the other approaches;
2. in general, TDF can be seen as an extension of TF that also takes into account the spread of terms across documents.

However, for the sake of completeness, and to confirm the findings in Reference [23], here, we also compare it to all the most recent approaches proposed in the literature and compatible with our constraints, and specifically: IDF, NIDF, TF*IDF, H, TRS, and NTF.

For direct comparison to the latest literature, we used the same dataset as in Reference [23], concerning Italian language. Since, as already noted, for single documents TDF boils down to TF, and approaches based on document frequency or distribution are not applicable to single documents, this comparison makes sense only on groups of texts. So, from the dataset in Reference [23], we considered the sets of texts PPI (as in this paper), NTT (Non Technical Texts), and All. We investigated performance for increasingly larger corpora, in the case of both homogeneous (PPI and NTT) and mixed-style (All) texts.

We evaluated performance in terms of Precision (P, the ratio of a set of candidate stopwords for a language that are actually stopwords in that language) and Recall (R, the ratio of actual stopwords for a language that are included in a set of candidate stopwords for a language). Not to make the evaluation dependent on a specific cutpoint, and in order to check whether there is a correlation between the ranking and the actual stopwordness, we actually computed $P@n$ and $R@n$, meaning that P and R were computed on the top n candidate stopwords in the ranking returned by the algorithms. Values of n were taken as a multiples of 10. We considered up to the first 100 candidate stopwords returned by each competitor ($n = 100$), except IDF, for which performance @100 could not be assessed. Indeed, due to the small number of documents, it was unable to provide a fine-grained ranking: the best score is shared by 2510 terms for PPI, by 382 terms for NTT, and by 160 terms for All.

Table 5 shows the Precision outcomes. We first note that TF*IDF, NIDF, H and TRS (on the All dataset) all show an undesirable non-monotonic behavior. This means that wrong stopwords are included in the very top items of the ranking, spoiling the performance of subsequent thresholds. This also means that the automatic cutoff method could not be applied reliably on these approaches, since it works best with monotonically decreasing performance. As expected, all approaches based on inverse document frequency or distribution (IDF, NIDF, TF*IDF, H) improve for increasingly large datasets. However, possibly due to the small number of texts, they are clearly the worst, with $P < 0.50$ (and often $P << 0.50$) for all datasets and number of candidates, except NIDF on All texts, but only for the very top candidate stopwords (up to $P@50$). In particular, albeit considering many more candidate terms, IDF is clearly and by far the worst approach ($P < 0.50$ always, even on the complete dataset 'All'). Being based on the distribution of terms across documents, H rewards even terms with very few occurrences, but evenly spread in the corpus. Note also that H is among the most computationally expensive approaches in the comparison, since it needs to count the frequency of each term in each single document. TRS is the non-TF-based approach with closest performance to TF-based ones but still far from them (interestingly, it performs worst on the complete dataset, which was somehow unexpected).

NTF and TF have very good performance, and very similar to each other: in some cases, NTF is slightly worse; in some others, it is slightly better, especially when considering more candidate stopwords (*P*@80–100). Both show very good performance, always above 0.90 up to *P*@50 and always above 0.72 up to *P*@100 (only once TF is below, for $P@100 = 0.69$ on NTT).

Notwithstanding the very good performance of TF and NTF, TDF is able to further improve over them, usually showing a smoother decay and higher precision. Only in 4 cases out of 30 (on the smallest dataset PPI @60 and @80–100) is it worse, and only @80 significantly (more than 0.02). This is probably due to the smaller number of texts, preventing good contribution from the DF component, and to the fact that, being the texts in PPI similar to each other (they are different volumes of the same work), they include many domain-specific words with very high frequency that are not stopwords. The top-ranked candidate stopwords are almost perfect (precision is 1.00 for all groups @30, and for 'All' even @40). A detailed analysis of the wrong stopwords returned by TDF reveals that most of them might be considered, however, domain-dependent stopwords (e.g., the abbreviation 'art.' for 'articolo'—i.e., 'law article'— in CCI). So, we are confident that it is possible to recognize 'meaningless' domain words, to be considered as stopwords for domain-specific applications.

Table 5. Precision of TDF compared to other approaches in the literature.

Group	Approach	P@10	P@20	P@30	P@40	P@50	P@60	P@70	P@80	P@90	P@100
PPI	IDF					0.06 (@2510)					
	TF*IDF	0.10	0.05	0.07	0.05	0.04	0.03	0.03	0.04	0.03	0.03
	NIDF	0.50	0.25	0.20	0.15	0.12	0.10	0.09	0.07	0.07	0.06
	H	0.00	0.00	0.10	0.15	0.20	0.22	0.26	0.24	0.24	0.26
	TRS	0.90	0.90	0.90	0.85	0.84	0.82	0.74	0.67	0.62	0.58
	NTF	1.00	1.00	0.97	0.95	0.94	0.90	0.86	0.83	0.78	0.72
	TF	1.00	1.00	0.97	0.95	0.94	0.90	0.86	0.83	0.78	0.72
	TDF	1.00	1.00	1.00	0.95	0.94	0.88	0.86	0.81	0.73	0.71
NTT	IDF					0.28 (@382)					
	TF*IDF	0.20	0.15	0.27	0.25	0.22	0.30	0.30	0.29	0.27	0.28
	NIDF	0.50	0.25	0.20	0.32	0.42	0.43	0.43	0.42	0.38	0.34
	H	0.00	0.10	0.07	0.05	0.10	0.13	0.17	0.19	0.18	0.17
	TRS	1.00	0.95	0.90	0.85	0.82	0.75	0.73	0.71	0.67	0.63
	NTF	1.00	1.00	0.97	0.95	0.90	0.87	0.86	0.80	0.76	0.72
	TF	1.00	1.00	1.00	0.97	0.92	0.90	0.86	0.79	0.74	0.69
	TDF	1.00	1.00	1.00	0.97	0.94	0.92	0.86	0.79	0.76	0.73
All	IDF					0.41 (@160)					
	TF*IDF	0.40	0.40	0.47	0.43	0.44	0.45	0.41	0.44	0.42	0.46
	NIDF	0.80	0.85	0.83	0.62	0.52	0.45	0.43	0.39	0.39	0.40
	H	0.40	0.45	0.37	0.35	0.34	0.28	0.27	0.25	0.24	0.26
	TRS	0.60	0.95	0.53	0.70	0.76	0.41	0.44	0.40	0.38	0.33
	NTF	1.00	1.00	0.97	0.95	0.90	0.88	0.81	0.80	0.74	0.73
	TF	1.00	1.00	1.00	0.95	0.92	0.88	0.84	0.80	0.74	0.72
	TDF	1.00	1.00	1.00	1.00	0.94	0.92	0.86	0.80	0.77	0.73

In spite of Recall performance being typically inverse to that of Precision, Table 6 confirms the same behavior also for Recall (here, we just report *R*@100). Again, only NTF and TF are somehow comparable to TDF, with recall values above 0.25. The performance of other approaches, even on all documents, is completely useless in practice. Again, TDF is able to provide significant improvements over TF and NTF. Note that, since the golden standard consists of 279 stopwords, the maximum *R*@100 for any possible approach on those (collections of) texts is actually $100/279 = 0.36$. So, TDF reaching values 0.28 and 0.31, can be considered a really impressive result.

Table 6. Recall @100 of TDF compared to other approaches in the literature

	PPI	**NTT**	**All**
IDF	0.58 (@2510)	0.39 (@382)	0.23 (@160)
TF*IDF	0.01	0.10	0.16
H	0.09	0.06	0.09
NIDF	0.03	0.14	0.14
TRS	0.21	0.16	0.12
NTF	0.26	0.26	0.26
TF	0.25	0.25	0.26
TDF	0.31	0.28	0.28

While it is impossible to formally prove in general the superiority of TDF against its competitors, a comparison of its precision with the baseline consisting of random selection of stopwords may give an idea of its discrimination power. Selecting k terms at random from a vocabulary of m terms including n stopwords, the likelihood that all the k selected terms are actually stopwords is given by

$$\prod_{i=0}^{k} \frac{n-i}{m-i}.$$

Indeed, we have n/m chances of selecting a keyword in the first choice, which must be combined in conjunction with the $(n-1)/(m-1)$ chances of selecting one of the $n-1$ remaining stopwords among the $m-1$ remaining terms in the second choice, and so on until the k-th choice. Now, let us consider the cases in Table 5 in which TDF reached 100% precision but its competitors did not, and compute the likelihood of obtaining such precision at random. Specifically, the cases are:

- P@30 on PPI: the corresponding likelihood is 2.39×10^{-66}.
- P@40 on All: the corresponding likelihood is 1.33×10^{-96}.

The odds are so small that we may consider it as a practical proof of reliability of the result and, thus, of the superiority of TDF over its competitors.

5.2. Cutoff Assessment Performance on Single Texts

Our second experiment analyzes the behavior of the automatic cutoff strategy on single texts, where TDF = TF. While the reader may analyze the figures in more detail and from different perspectives, here, we will comment on some aspects that we consider more relevant.

Table 7 shows the results of T(D)F on the English corpus. Again, we used Precision P and Recall R, computed on the set of candidate stopwords returned by our algorithm using the automatic cutpoint assessment approach. As expected, the worst performance from almost all perspectives is for D33 because it uses a less varied vocabulary and a technical language. So, we will not discuss it further. Note from Table 1 that basically each single text includes almost all stopwords in the golden standard, which makes the problem quite challenging because the target recall R_{max} is very high. As expected, performance is in general proportional to the length of the text, and always very high ($P + R$ being always $\geq$1, except for TBA with $a = -1$ and $a = -0.5$, where it is slightly below 1.0). Even for shorter texts (the shortest one being DJaMH), precision is very high up to the loosest setting ($a = -0.25$): sometimes 1.0, always >0.81. The best performance is obtained on Scw, which is the longest text but, surprisingly, not using a narrative style and in archaic language.

Table 8 shows the performance on Italian texts. Here, with a couple of exceptions (IPS and TlN, which are the longest narrative texts), R_{max} is less than 0.61 (see Table 2). The worst performance, as expected, is obtained on HeG, which is actually extremely short. For this very short text, the effect of reducing the $-a$ parameter is much more emphasized than for longer texts. Lower performance is obtained on non-narrative texts, but still with $P + R \geq 0.75$ in all settings. Compared to English, precision is still very high also in the loosest setting ($a = -0.25$), but recall is lower than for English (perhaps due to the texts being significantly shorter than those in the English corpus and to the maximum recall reachable being much less than for English texts).

Table 7. Performance of cutoff assessment on single English texts.

a	-1.0					-0.5					-0.25				
ID	**#*c***	**#*s***	***P***	***R***	***P* + *R***	**#*c***	**#*s***	***P***	***R***	***P* + *R***	**#*c***	**#*s***	***P***	***R***	***P* + *R***
DJaMH	16	16	1.00	0.08	1.08	16	16	1.00	0.08	1.08	26	16	1.00	0.13	1.13
CC	19	19	1.00	0.09	1.09	32	30	0.94	0.15	1.09	32	30	0.94	0.15	1.09
F	16	16	1.00	0.08	1.08	36	36	1.00	0.18	1.18	36	36	1.00	0.18	1.18
TBA	18	16	0.89	0.08	0.97	20	18	0.90	0.09	0.99	38	34	0.89	0.17	1.06
ACYiKAC	12	12	1.00	0.06	1.06	26	26	1.00	0.13	1.13	45	45	1.00	0.22	1.22
M-D	21	21	1.00	0.10	1.10	36	35	0.97	0.17	1.14	48	47	0.97	0.18	1.15
TCoMC	35	35	1.00	0.17	1.17	35	35	1.00	0.17	1.17	67	59	0.88	0.29	1.17
TTM	14	14	1.00	0.07	1.07	45	43	0.96	0.21	1.17	65	55	0.85	0.27	1.12
TYA	17	17	1.00	0.08	1.08	44	41	0.93	0.20	1.13	65	55	0.85	0.27	1.12
TVdB	19	19	1.00	0.09	1.09	39	36	0.92	0.18	1.10	50	47	0.94	0.23	1.17
LdlV	23	23	1.00	0.11	1.11	35	33	0.94	0.16	1.10	73	60	0.82	0.30	1.12
TYL	19	19	1.00	0.09	1.09	47	44	0.94	0.22	1.16	64	58	0.91	0.29	1.20
TMitIM	18	18	1.00	0.09	1.09	35	33	0.94	0.16	1.10	59	52	0.88	0.26	1.14
Scw	57	48	0.84	0.24	1.08	91	75	0.82	0.37	1.19	111	90	0.81	0.45	1.26
D33	16	12	0.75	0.06	0.81	31	19	0.61	0.09	0.70	41	23	0.56	0.11	0.67

Table 8. Performance of cutoff assessment on single Italian texts.

a	-1.0					-0.5					-0.25				
ID	**#*c***	**#*s***	***P***	***R***	***P* + *R***	**#*c***	**#*s***	***P***	***R***	***P* + *R***	**#*c***	**#*s***	***P***	***R***	***P* + *R***
PPI1	23	23	1.00	0.08	1.08	31	31	1.00	0.11	1.11	41	39	0.95	0.14	1.09
PPI2	19	19	1.00	0.07	1.07	31	29	0.94	0.10	1.04	41	39	0.95	0.14	1.09
PPI3	20	20	1.00	0.07	1.07	29	27	0.93	0.10	1.03	50	45	0.90	0.16	1.06
PPI4	22	22	1.00	0.08	1.08	31	31	1.00	0.11	1.11	40	37	0.93	0.13	1.06
PPI5	20	20	1.00	0.07	1.07	36	34	0.94	0.12	1.06	39	37	0.95	0.13	1.08
L'E	23	22	0.96	0.08	1.05	40	34	0.85	0.12	0.97	40	34	0.85	0.12	0.97
IPS	35	32	0.91	0.11	1.02	37	33	0.89	0.12	1.01	54	43	0.80	0.15	0.95
TlN	31	30	0.97	0.11	1.08	52	47	0.90	0.17	1.07	52	47	0.90	0.17	1.07
HeG	5	5	1.00	0.02	1.02	18	14	0.78	0.05	0.83	171	68	0.40	0.24	0.64
LDC	17	15	0.88	0.05	0.93	37	30	0.81	0.11	0.92	56	40	0.71	0.14	0.85
CCI	38	33	0.87	0.12	0.99	57	41	0.72	0.15	0.87	66	41	0.62	0.15	0.75

Table 9 reports performance on Squinzanese texts. All the texts in this corpus are quite short (see Table 3) and include a very small portion of the stopwords in the golden standard, which has clear consequences on performance, and especially on recall. As for HeG in the Italian corpus, for shorter texts taking $a = -0.25$ causes a significant increase in the number of selected candidate stopwords, with no improvement in the quality of the candidates. Nevertheless, in most of the cases, we may consider performance to be satisfactory, given the very challenging problem.

For the sake of comparison, we may take as a baseline the approach of cutting the list of candidate stopwords at the position where the largest weight gap between adjacent items in the ranking occurs. Table 10 shows, for each language (first column), which single texts (fourth column) and groups of texts (fifth column) returned the number of stopwords (#*s*) reported in the second column using this method. It is evident that the method is useless: in most cases, it returns just 1 stopword. Only for Italian and Squinzanese does it sometimes return more than 2 stopwords, exceptionally returning more than 3 for some Squinzanese texts. Interestingly, groups of texts do not improve performance over single texts. This behavior is not surprising, since, given the typical weight plots as in Figure 1, with a very steep decay on the left, the largest difference happens very early in the ranking. Since TDF always places correct stopwords in the top positions of the ranking, Precision is always 1 for these few stopwords. For the reader's reference, the third column in Table 10 reports the recall (*R*) corresponding to each given number of stopwords for each language.

Table 9. Performance of cutoff assessment on single Squinzanese texts.

a			**−1.0**					**−0.5**					**−0.25**		
ID	**#*c***	**#*s***	***P***	***R***	***P*** **+** ***R***	**#*c***	**#*s***	***P***	***R***	***P*** **+** ***R***	**#*c***	**#*s***	***P***	***R***	***P*** **+** ***R***
LFF	12	10	0.83	0.05	0.88	16	14	0.88	0.07	0.95	167	63	0.38	0.34	0.72
MFF	16	12	0.75	0.06	0.81	16	12	0.75	0.06	0.81	26	21	0.81	0.11	0.92
LMS	8	8	1.00	0.04	1.04	12	10	0.83	0.05	0.88	92	39	0.42	0.21	0.63
LNN	13	11	0.85	0.06	0.91	15	12	0.80	0.06	0.86	25	21	0.84	0.11	0.95
LR	11	11	1.00	0.06	1.06	13	12	0.92	0.06	0.98	20	15	0.75	0.08	0.83
LN	8	8	1.00	0.04	1.04	17	11	0.65	0.06	0.71	92	42	0.46	0.22	0.68
LCS	9	9	1.00	0.05	1.05	15	11	0.73	0.06	0.79	140	57	0.41	0.31	0.72
LPZ	9	9	1.00	0.05	1.05	11	9	0.82	0.05	0.87	88	43	0.49	0.23	0.72
LJ	8	8	1.00	0.04	1.04	19	17	0.89	0.09	0.98	122	57	0.47	0.30	0.77
BclNeRclS	12	12	1.00	0.06	1.06	15	13	0.87	0.07	0.94	26	20	0.77	0.11	0.88
ABA	7	7	1.00	0.04	1.04	15	13	0.87	0.07	0.94	75	38	0.51	0.20	0.71
MS	13	11	0.85	0.06	0.91	13	11	0.85	0.06	0.91	26	23	0.88	0.12	1.00
RF	13	10	0.77	0.05	0.82	14	11	0.79	0.06	0.85	34	27	0.79	0.14	0.93
II	10	10	1.00	0.05	1.05	16	14	0.88	0.07	0.95	23	18	0.78	0.10	0.88
RS	17	13	0.76	0.07	0.83	17	13	0.76	0.07	0.83	27	20	0.74	0.11	0.85
LT	6	6	1.00	0.03	1.03	15	12	0.80	0.06	0.86	17	12	0.71	0.06	0.77
LV	10	10	1.00	0.05	1.05	10	10	1.00	0.05	1.05	95	40	0.42	0.21	0.66
LT2	11	10	0.91	0.05	0.96	11	10	0.91	0.05	0.96	125	59	0.47	0.31	0.78

Table 10. Performance of baseline cutoff assessment.

Language	**#*s***	***R***	**Single Texts**	**Groups of Texts**
English	1	0.00	DJaMH, CCE, F, TBA, MD, TCoMC, TTM, TYA, TVdB, LdlV, TYL, TMitIM, d33	Novels, Dumas, Tech, noTech
	2	0.01	ACYiKAC, Scw	—
Italian	1	0.00	PPI1, PPI2, PPI3, PPI4, TlN, HeG, CCI	PPI, Novels
	2	0.01	PPI5, LDC	Tech, noTech, All
	3	0.01	IPS, L'E	—
Squinzanese	1	0.01	MFF, LMS, LNN, LR, LCS, LPZ, ABA, MS, RF, RS, LV	1–6, 7–12, 13–18, 1–12, All
	2	0.01	LN	—
	3	0.02	LJ, BclNeRclS, LT	—
	6	0.03	II	—
	7	0.04	LFF	—
	8	0.04	LT2	—

5.3. Performance on Sets of Texts

Given the good performance of the cutoff threshold assessment and of TDF on Italian compared to the state-of-the-art, reported in previous subsections, we now focus on the joint performance of TDF and of the cutoff threshold assessment on different languages. Table 11 shows the experimental results for the various languages at various values of the slope parameter a.

The worst precision in the entire table is 0.66, which ensures applicability of the approach to very different languages and experimental settings. As expected, for all languages, the maximum number of correct stopwords is obtained for the loosest slope ($a = -0.25$) and for the complete set of texts ('All'). It is worth noting that the number of stopword selected for English and Italian in this setting (171 and 122, respectively) is surprisingly close to the number of stopwords of these languages, as assessed in Reference [39] (174 and 134, respectively). Except for Squinzanese with $a = -1$ on group 'All', the number of candidate and correct stopwords retrieved increases for larger groups of texts. Interestingly, for all groups of texts and languages, the rate of increase in number of returned candidate stopwords for smaller $|a|$ is much more than the corresponding rate of decrease in precision. The most cautious slope setting ($a = -1$) ensures very high precision, obviously at the expenses of recall. The loosest slope setting ($a = -0.25$) doubles recall without dramatically dropping precision. In fact, $P + R$ is quite stable, and always >1.00, except for some cases in Italian (which is a complex language). English achieves the best results in all metrics for all values of the slope parameter (albeit, of course, the datasets in different languages are not comparable), consistently with its being the easiest case (syntactically simpler and with more text in the dataset). Especially recall is very high (0.62) on 'All' with slope $a = -0.25$. Squinzanese retrieved the smallest number of stopwords, but it was the more complex case (dataset with less text and including the smallest percentage of stopwords in the golden standard). Partly unexpected, technical texts ('Tech') reach good performance both on English and on Italian, comparable to that of narrative texts or even better (in English for looser slopes), notwithstanding their shorter texts and more peculiar styles.

Table 11. TDF outcomes on sets of texts.

a			**−1.0**					**−0.5**					**−0.25**		
ID	**#*c***	**#*s***	***P***	***R***	***P* + *R***	**#*c***	**#*s***	***P***	***R***	***P* + *R***	**#*c***	**#*s***	***P***	***R***	***P* + *R***
English															
Novels	39	39	1.00	0.19	1.19	46	46	1.00	0.23	1.23	80	74	0.93	0.37	1.30
Dumas	62	56	0.92	0.28	1.20	74	65	0.88	0.32	1.20	87	75	0.86	0.37	1.23
Tech	44	42	0.95	0.21	1.16	73	67	0.92	0.33	1.25	108	87	0.81	0.44	1.25
noTech	64	60	0.94	0.30	1.24	84	77	0.92	0.38	1.30	117	98	0.84	0.49	1.33
All	67	63	0.94	0.31	1.25	112	99	0.88	0.49	1.37	171	125	0.73	0.62	1.35
Italian															
PPI	35	34	0.97	0.12	1.09	53	49	0.92	0.18	1.10	61	55	0.90	0.20	1.10
Novels	36	33	0.92	0.12	1.04	67	51	0.76	0.18	0.94	67	51	0.76	0.18	0.94
Tech	38	33	0.87	0.12	0.99	52	40	0.77	0.14	0.91	75	51	0.68	0.18	0.86
noTech	48	44	0.92	0.16	1.08	81	64	0.79	0.23	1.02	118	80	0.68	0.29	0.97
All	58	53	0.91	0.19	1.10	87	68	0.78	0.24	1.02	122	81	0.66	0.29	0.95
Squinzanese															
1-6	17	16	0.94	0.08	1.04	33	29	0.88	0.15	1.03	47	38	0.81	0.20	1.01
7-12	19	19	1.00	0.10	1.10	24	23	0.96	0.12	1.08	41	37	0.90	0.20	1.10
13-18	16	16	1.00	0.08	1.08	27	24	0.89	0.13	1.02	54	43	0.80	0.23	1.03
1-12	26	24	0.92	0.13	1.05	38	34	0.89	0.18	1.07	59	45	0.76	0.24	1.00
All	23	21	0.91	0.11	1.02	43	36	0.84	0.19	1.03	63	48	0.76	0.25	1.01
Multilingual — All (English) ∪ All (Italian)															
	93	90	0.97	0.18	1.15	137	123	0.90	0.26	1.15	191	155	0.81	0.32	1.13

Very interesting is the multilingual case, run on the merger of all English and all Italian texts (and of their golden standards). For all settings of parameter a, the results of precision and of number of candidate stopwords (overall and correct) retrieved is larger than those of the 'All' datasets for the single languages. This was not obvious, since merging different languages might spoil the frequencies of each of them and make their stopwords not recognizable. On the contrary, this result shows that, even for corpora that mix several languages, our proposed approach may be effective. In addition, recall is consistent with what was obtained on the single languages, in spite of the nearly doubled number of stopwords in the golden standard.

6. Conclusions

Many languages lack the linguistic resources needed by Natural Language Processing (NLP) approaches because they are language-specific, and manually building them is difficult. In particular, Stopword lists, i.e., lists of terms not carrying significant information for the texts in a corpus, are fundamental to improve effectiveness and efficiency of many tasks (Information Retrieval or Diachronic Analysis applications, linguistic studies, etc.). This paper concerned the automatic extraction of stopword lists from plain texts, proposing (i) a simple frequency-based approach to assign a degree of 'stopwordness' to each term in a corpus, and (ii) a geometric strategy to automatically determine the cutoff point in the ranking of candidate stopwords. Specifically, it focused on the case of very small corpora and of independence from language.

Extensive experiments have shown that both strategies (ranking and cut-off) are effective. Both proved to be effective on different languages and under different experimental settings (amount of training text, different styles, mixed languages). In particular, the approach for cutoff assessment is based on a very intuitive parameter that can be set by the experimenter, instead of the fixed strategies used so far in the literature, that may not be consistent with the frequency values of candidate stopwords. The stopword weighting approach is very simple; still, it significantly outperforms all comparable state-of-the-art solutions. Given these outcomes, we plan to investigate possible applications of the TDF techniques to other NLP tasks that may be carried out statistically, perhaps real-life scenarios of language modeling or machine translation.

Funding: This research received no external funding.

Data Availability Statement: All the data were taken from publicly available Internet websites, as specified in the text, except for those concerning the Squinzanese dialect. These data are copyrighted and cannot be made publicly available, but the authors are available to provide them to interested researchers under non-disclosure agreement.

Acknowledgments: The author would like to thank Giovanni Luca Izzi for his contribution in developing the initial version of this manuscript, and for the useful discussions about the TDF approach.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Miller, G.A. WordNet: A lexical database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
2. Vossen, E.P. (Ed.) *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*; Kluwer: Dordrecht, The Netherlands, 1998.
3. Ciravegna, F.; Magnini, B.; Pianta, E.; Strapparava, C. *A Project for the Construction of an Italian Lexical Knowledge Base in the Framework of WordNet*; Technical Report # 9406-15; IRST: Povo, Italy, 1994.
4. Kaur, J.; Buttar, P.K. A Systematic Review on Stopword Removal Algorithms. *Int. J. Future Revolut. Comput. Sci. Commun. Eng.* **2018**, *4*, 207–210.
5. Luhn, H.P. Keyword-in-context index for technical literature (kwic index). *J. Assoc. Inf. Sci. Technol.* **1960**, *11*, 288–295.
6. Tahmasebi, N.; Borin, L.; Jatowt, A. Survey of Computational Approaches to Lexical Semantic Change. *arXiv* **2018**, arXiv:1811.06278.
7. Haddi, E.; Liu, X.; Shi, Y. The Role of Text Pre-processing in Sentiment Analysis. *Procedia Comput. Sci.* **2013**, *17*, 26–32.
8. Saif, H.; Fernandez, M.; He, Y.; Alani, H. On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), Reykjavik, Iceland, 26–31 May 2014; pp. 810–817.

9. Ferilli, S.; Esposito, F. On Frequency-Based Approaches to Learning Stopwords and the Reliability of Existing Resources—A Study on Italian Language. In *Digital Libraries and Multimedia Archives. IRCDL 2018*; Communications in Computer and Information Science; Springer: Berlin/Heidelberg, Germany, 2018; Volume 806, pp. 69–80.
10. Wilbur, W.J.; Sirotkin, K. The Automatic Identification of Stop Words. *J. Inf. Sci.* **1992**, *18*, 45–55. [CrossRef]
11. Makrehchi, M.; Kamel, M.S. Automatic Extraction of Domain-Specific Stopwords from Labeled Documents. In *IR Research, Proceedings of the 30th European Conference on Advances in Information Retrieval,Glasgow, UK, 30 March–3 April 2008*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 222–233.
12. Makrehchi, M.; Kamel, M.S. Extracting Domain-specific Stopwords for Text Classifiers. *Intell. Data Anal.* **2017**, *21*, 39–62. [CrossRef]
13. Sinka, M.P.; Corne, D.W. Evolving Better Stoplists for Document Clustering and Web Intelligence. In *Design and Application of Hybrid Intelligent Systems*; IOS Press: Amsterdam, The Netherlands, 2003; pp. 1015–1023.
14. Popova, S.; Krivosheeva, T.; Korenevsky, M. Automatic Stop List Generation for Clustering Recognition Results of Call Center Recordings. In *Speech and Computer*; Ronzhin, A., Potapova, R., Delic, V., Eds.; Springer International Publishing: Berlin/Heidelberg, Germany, 2014; pp. 137–144.
15. Yaghoub-Zadeh-Fard, M.; Minaei-Bidgoli, B.; Rahmani, S.; Shahrivari, S. PSWG: An automatic stop-word list generator for Persian information retrieval systems based on similarity function POS information. In Proceedings of the 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, Iran, 5–6 November 2015; pp. 111–117.
16. Fox, C. A Stop List for General Text. *SIGIR Forum* **1989**, *24*, 19–21. [CrossRef]
17. Savoy, J. A stemming procedure and stopword list for general French corpora. *J. Assoc. Inf. Sci. Technol.* **1999**, *50*, 944–952. [CrossRef]
18. Sparck-Jones, K. A statistical interpretation of term specificity and its application in retrieval. *J. Doc.* **1972**, *28*, 11–21. [CrossRef]
19. Robertson, S.E.; Sparck-Jones, K. Relevance weighting of search terms. *J. Am. Soc. Inf. Sci.* **1976**, *27*, 129–146. [CrossRef]
20. Lo, R.T.W.; He, B.; Ounis, I. Automatically Building a Stopword List for an Information Retrieval System. *J. Digit. Inf. Manag.* **2005**, *5*, 17–24.
21. Cover, T.M.; Thomas, J.A. *Elements of Information Theory*; John Wiley: Hoboken, NJ, USA, 1991.
22. Ferilli, S.; Esposito, F.; Grieco, D. Automatic Learning of Linguistic Resources for Stopword Removal and Stemming from Text. *Procedia Comput. Sci.* **2014**, *38*, 116–123. [CrossRef]
23. Ferilli, S.; Izzi, G.L.; Franza, T. Automatic Stopwords Identification from Very Small Corpora. In *Intelligent Systems in Industrial Applications*; Studies in Computational Intelligence; Springer: Berlin/Heidelberg, Germany, 2020; p. 15.
24. Al-Shalabi, R.; Kanaan, G.; Jaam, J.M.; Hasnah, A.; Hilat, E. Stop-word removal algorithm for Arabic language. In Proceedings of the 2004 International Conference on Information and Communication Technologies: From Theory to Applications, Damascus, Syria, 23 April 2004; pp. 545–549.
25. Alhadidi, B.; Alwedyan, M. Hybrid Stop-Word Removal Technique for Arabic Language. *Egypt. Comput. Sci. J.* **2008**, *30*, 35–38.
26. El-Khair, I.A. Effects of Stop Words Elimination for Arabic Information Retrieval: A Comparative Study. *arXiv* **2017**, arXiv:1702.01925.
27. Zou, F.; Wang, F.L.; Deng, X.; Han, S.; Wang, L.S. Automatic Construction of Chinese Stop Word List. In Proceedings of the 5th WSEAS International Conference on Applied Computer Science, Hangzhou, China, 16–18 April 2006; World Scientific and Engineering Academy and Society (WSEAS): Stevens Point, WI, USA, 2006; pp. 1009–1014.
28. Zou, F.; Wang, F.L.; Deng, X.; Han, S. Evaluation of Stop Word Lists in Chinese Language. In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'06), Genoa, Italy, 22–28 May 2006; European Language Resources Association (ELRA): Paris, France, 2006; pp. 2504–2507.
29. Raulji, J.K.; Saini, J.R. Stop-Word Removal Algorithm and its Implementation for Sanskrit Language. *Int. J. Comput. Appl.* **2016**, *150*, 15–17.
30. Rakholia, R.M.; Saini, J.R. A Rule-Based Approach to Identify Stop Words for Gujarati Language. In *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications*; Satapathy, S.C., Bhateja, V., Udgata, S.K., Pattnaik, P.K., Eds.; Springer: Berlin/Heidelberg, Germany, 2017; pp. 797–806.
31. Puri, R.; Bedi, R.; Goyal, V. Automated Stopwords Identification in Punjabi Documents. *Eng. Sci. An Int. J.* **2013**, *8*, 119–125.
32. Garg, U.; Goyal, V. Effect of Stop Word Removal on Document Similarity for Hindi Text. *Eng. Sci. An Int. J.* **2014**, *2*, 3.
33. Jha, V.; Manjunath, N.; Shenoy, P.D.; Venugopal, K.R. HSRA: Hindi stopword removal algorithm. In Proceedings of the 2016 International Conference on Microelectronics, Computing and Communications (MicroCom), Durgapur, India, 23–25 January 2016; pp. 1–5.
34. Siddiqi, S.; Sharan, A. Construction of a generic stopwords list for Hindi language without corpus statistics. *Int. J. Adv. Comput. Res.* **2017**, *8*, 35–40. [CrossRef]
35. Pan, S.; Saha, D. An automatic identification of function words in TDIL tagged Bengali corpus. *Int. J. Comput. Sci. Eng.* **2019**, *7*, 20–27.
36. Wijeratne, Y.; de Silva, N. Sinhala Language Corpora and Stopwords from a Decade of Sri Lankan Facebook.*arXiv* **2020**, arXiv: 2007.07884.

37. Rajkumar, N.; Subashini, T.S.; Rajan, K.; Ramalingam, V. Tamil Stopword Removal Based on Term Frequency. In *Data Engineering and Communication Technology*; Raju, K.S., Senkerik, R., Lanka, S.P., Rajagopal, V., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; pp. 21–30.
38. Messito, A. *Cuntame Nnu Cuntu!* Photocity: Napoli, Italy, 2014.
39. Saini, J.R.; Rakholia, R.M. On Continent and Script-Wise Divisions-Based Statistical Measures for Stop-words Lists of International Languages. *Procedia Comput. Sci.* **2016**, *89*, 313–319. [CrossRef]