text2sdg: An R Package to Monitor Sustainable Development Goals from Text

by Dominik S. Meier, Rui Mata, and Dirk U. Wulff

Abstract Monitoring progress on the United Nations Sustainable Development Goals (SDGs) is important for both academic and non-academic organizations. Existing approaches to monitoring SDGs have focused on specific data types; namely, publications listed in proprietary research databases. We present the text2sdg package for the R language, a user-friendly, open-source package that detects SDGs in text data using different individual query systems, an ensemble of query systems, or custom-made ones. The text2sdg package thereby facilitates the monitoring of SDGs for a wide array of text sources and provides a much-needed basis for validating and improving extant methods to detect SDGs from text.

1 Introduction

The United Nations Sustainable Development Goals (SDGs) have become an important guideline for both governmental and non-governmental organizations to monitor and plan their contributions to social, economic, and environmental transformations. The 17 SDGs cover large areas of application, from ending poverty and improving health, to fostering economic growth and preserving natural resources. As the latest UN report (UN, 2022) attests, the availability of high-quality data is still lacking in many of these areas and progress is needed in identifying data sources that can help monitor work on these goals. Monitoring of SDGs has typically been based on economic and health data, which are often difficult and costly to gather (e.g., `https://sdg-tracker.org/`; `https://www.sdgindex.org/`). One attractive alternative that has emerged from recent scientometric efforts is to detect SDGs from text, such as academic publications. Digitized text represents an attractive resource for monitoring SDGs across a large number of domains because it is becoming widely available in various types of documents, such as news articles, websites, corporate reports, and social media posts. In light of this promise, we developed **text2sdg**, a freely available, open-source tool to enable the SDG-labeling of digitized text and facilitate methodological development in this area. In what follows, we first present some background on existing labeling systems developed to identify SDGs from text, and then provide an overview of the **text2sdg** package, showcase its use in a representative case study, and discuss the promise and limitations of the approach.

2 An overview of SDG labeling systems

The **text2sdg** package provides a user-friendly way to use any existing or custom-made labeling system developed to monitor the 17 SDGs in text sources. The package implements six different labeling systems utilizing different keywords and keyword combination rules, as well as an ensemble model based on the six systems that was trained on labeled data. In the following, we will first introduce the six existing labeling systems, namely the Elsevier, Aurora, Auckland, SIRIS, SDGO, and SDSN systems, before discussing how these systems are combined within the ensemble approach. See table 1 for overview of these labeling systems. We address custom-made labeling systems in a dedicated section below.

2.1 Individual labeling systems

The most prominent SDG labeling system has been developed by *Elsevier*. The Elsevier labeling system was integrated into the Times Higher Education Impact Rankings in 2019, which at the time compared 1,118 universities in their efforts to address the SDGs as measured by the frequency of SDG-related terms in their academic output. The Elsevier queries consist of a list of expert-vetted keywords that are combined using logical AND operators, implying that multiple keywords must be met to label a document as containing a certain SDG. The development of the queries started with an original list of keywords for each SDG that were iteratively fine tuned to maximize the number of identified papers closely reflecting the different SDGs. This involved cropping or combining keywords to reduce the number of irrelevant hits. A detailed report on the initial development of the Elsevier query system is provided by Jayabalasingham et al. (2019). Since the first version, the Elsevier labeling system has been iteratively improved, with the latest versions including additional information specific to academic publications and the Scopus database, such as identifiers of journal names or research areas.

text2sdg implements the latest version without such additional identifiers to broaden the package's applicability beyond the Scopus database (Jayabalasingham et al., 2019).

The Aurora Universities Network's "Societal Impact and Relevance of Research" working group started to develop a labeling system in 2017 to increase the visibility of research into the SDGs. Aurora's queries were developed with the goal of identifying SDG-related academic publications included in the Scopus database. Consequently, the syntax of Aurora queries is similar to the Scopus query language and the Elsevier system. However, in contrast to the Elsevier system, the queries combine keywords in a more complex fashion, recruiting Boolean (AND, OR) and proximity operators (e.g., w/3, implying within 3 words). As a result, Aurora's keywords are more specific, possibly leading to a smaller number of false positives. The initial version of the Aurora system only included terms that appear in the SDG policy text of the targets and indicators defined by the United Nations. Subsequent versions expanded on this by including additional keywords that reflect academic terminology. **text2sdg** implements version 5.0 of the Aurora labeling system (Vanderfeesten et al., 2020a). This version represents an improvement on previous versions based on a survey study (Vanderfeesten et al., 2020b) and modifications inspired in other efforts, namely those from Elsevier (above) and SIRIS (introduced below).

The Auckland labeling system (Wang et al., 2023) was developed by the University of Auckland to better understand how their research output contributes to the SDGs. To construct the queries, they used text-mining techniques to extract global and local SDG keywords from publication metadata. These keywords were then sorted according to the number of publications that include the terms and according to the keywords' term frequency–inverse document frequency. The top-ranked keywords were then manually reviewed to only retain keywords that are relevant. The selected keywords were then combined with those of SDSN and Elsevier as well as UN SDG Indicators to form the final SDG keyword list. These queries formed the basis for the Auckland queries, which make use of Boolean (AND, OR) operators and wildcards (e.g., "*").

The SIRIS labeling system (Duran-Silva et al., 2019) was created by SIRIS Academic as part of the "science4sdgs" project to better understand how science, innovation efforts, and technology related to the SDGs. The SIRIS queries were constructed in a five-step procedure. First, an initial list of keywords was extracted from the United Nations official list of goals, targets and indicators. Second, the list was manually enriched on a basis of a review of SDG relevant literature. Third, a word2vec model that was trained on a text corpus created from the enriched keyword list was used to identify keywords that were semantically related to the initial list. Fourth, using the DBpedia API, keywords were added that, according to the Wikipedia corpus, had a categorical relationship with the initial list. Fifth, and finally, the keyword list was manually revised. The queries of the SIRIS labeling system primarily consist of individual keywords that occasionally are combined with a logical AND. **text2sdg** implements the only currently available version of the SIRIS labeling system (Duran-Silva et al., 2019) .

The Open Source SDG (OSDG) project combines data from multiple sources to detect SDGs in text. Instead of developing yet another query system, OSDG's aim was to re-use and integrate existing knowledge by combining multiple SDG "ontologies" (i.e., query systems). OSDG has also made use of Microsoft Academic Graph to improve their results but because our query-based system cannot implement this procedure, we adopt the simpler ontology initially proposed by OSDG, which we refer to as "SDGO" in the package. The labeling system was based on central keywords in the SDG United Nations description (e.g."sanitation" was classified into "SDG6") and then manually expanded with additional relevant keywords identified from a corpus of already labeled documents. The resulting keyword list only makes use of the OR operator. **text2sdg** implements the only currently available version of these queries (Bautista, 2019).

Finally, the Sustainable Development Solutions Network (SDSN, Sustainable Development Solutions Network (SDSN), 2021) labeling system contains SDG-specific keywords compiled in a collaborative effort by several universities from the Sustainable Development Solutions Network (SDSN) Australia, New Zealand & Pacific Network. This query system was developed to detect SDGs in large sets of university-related text data, such as course listings or research publications. The authors used United Nations documents, Google searches, and personal communications as sources for the keywords. This query system combines keywords with OR operators and does not make use of AND operators.

All in all, as can be seen in Table 1, the latter systems differ from the former four in the complexity of their queries: the Elsevier, Aurora, Auckland, and SIRIS systems make use of keyword-combination queries and other criteria, such as proximity operators, whereas SDGO and SDSN only make use of keywords.

Labeling system	SDGs covered	Query operators	Unique keywords per SDG (mean & SD)	Example query (SDG-01)
Elsevier	SDG 1 - SDG 16	OR, AND, wildcards	74.9 (21.7)	"extreme poverty"
Aurora	SDG 1 - SDG 17	OR, AND, wildcards, proximity search	89.6 (31.6)	("poverty") W/3 ("chronic*" OR "extreme")
Auckland	SDG 1 - SDG 16	OR, AND, wildcards	183 (46.5)	"poverty eradication"
SIRIS	SDG 1 - SDG 16	OR, AND	262 (148)	("anti-poverty") AND ("poverty" OR "vulnerability")
SDGO	SDG 1 - SDG 17	OR	245 (236)	"absolute poverty"
SDSN	SDG 1 - SDG 17	OR	62.6 (16.8)	"End poverty"

Table 1: Overview of the labeling systems implemented in **text2sdg**. Legend: OR—keywords are combined using logical ORs, implying that only the keywords must be matched to assign an SDG label; AND—keywords are combined using logical ANDs, implying that multiple keywords must be matched to assign an SDG label; wildcards—keywords are matched considering different keyword parts; proximity search—keywords must co-occur within a certain word window to assign an SDG label.

2.2 The ensemble labeling system

In another publication (Wulff et al., 2023), we evaluated the accuracy of the six labeling systems implemented by **text2sdg** and a rival approach (i.e., OSDG Pukelis et al., 2020) using expert-labeled data sets. These analyses lead to three critical observations. First, the accuracy of SDG classifications was reasonable for all systems, but varied considerably as a function of the data set. This is because the systems differ in how liberal or conservative they assign SDGs to texts due to differences in the types of query operators they employ. Specifically, employing only OR-operators, SDGO and SDSN were considerably more liberal, whereas the other four systems employing additional operators were more conservative. In other words, the systems implement different trade-offs between sensitivity (i.e., true-positive rate) and specificity (i.e., true-negative rate). As a result, SDGO and SDSN outperformed the other systems for SDG-rich documents and vice versa. In addition to these differences in accuracy, we observed critical biases in SDG profiles, with the systems overemphasizing different sets of SDGs, and strong dependencies between SDG predictions and document length. To address these limitations, we developed an ensemble model approach that uses the the predictions of the six systems and document length as inputs to a random forest model. After training with expert-labeled and synthetic data, the ensemble model showed better out-of-sample accuracy, lower false alarm rates, and smaller biases than any individual labeling system Wulff et al. (2023). As a result, this ensemble model is also made available through **text2sdg** using a dedicated function.

In the following sections, we provide an overview over the **text2sdg** R package and demonstrate how its functions can be used to run to detect and analyze SDGs in text.

3 The text2sdg package

3.1 Motivation for text2sdg

Despite the effort put into developing various labeling systems and their great promise in addressing the SDG-related data scarcity, extant implementations of these approaches are not without shortcomings. First, the labeling systems were mostly developed to be used within academic citation databases (e.g., Scopus) and are not easily applied to other text sources. Second, existing implementations lack transparent ways to communicate which features are matched to which documents or how they compare between a choice of labeling systems. We alleviate these shortcomings by providing an open-source solution, **text2sdg**, that lets users detect SDGs in any kind of text using any of the above-mentioned systems, and ensemble of systems, or even customized, user-made labeling systems. The package provides a common framework for implementing the different extant or novel approaches and makes it easy to quantitatively compare and visualize their results.

3.2 Overview of text2sdg package

At the heart of the **text2sdg** package are the Lucene-style queries that are used to detect SDGs in text and the ensemble models that build on these queries. The queries map text features (i.e., words or a combination of words) to SDGs. For example, a text that contains the words "fisheries" and

Function Name	Description
`detect_sdg`	identifies SDGs in text using an ensemble model that draws on the six labeling systems (Elsevier, Aurora, Auckland, SIRIS, SDGO, SDSN).
`detect_sdg_systems`	identifies SDGs in text by using labeling systems (Elsevier, Aurora, Auckland, SIRIS, SDGO, SDSN).
detect_any	similar to `detect_sdg` but identifies SDGs in text using user-defined queries.
`crosstab_sdg`	crosstab_sdg takes the output of detect_sdg, detect_sdg_systems, or detect_any as input and determines correlations between either query systems or SDGs.
`plot_sdg`	takes the output of detect_sdg, detect_sdg_systems, or detect_any as input and produces adjustable barplots illustrating the hit frequencies produced by the different query systems.

Table 2: Overview of package functions

"marine" would be mapped to SDG 14 (i.e., conserve and sustainably use the oceans, seas and marine resources for sustainable development) by the Aurora system. To enable the use of such queries in R, the **text2sdg** package recruits the **corpustools** package (Welbers and van Atteveldt, 2021). **corpustools** has been built to implement complex search queries and execute them efficiently for large amounts of text. Based on this, **text2sdg** provides several functions that implement extant labeling systems, facilitate the specification of new labeling systems, and analyze and visualize search results. Table 2 gives an overview of the **text2sdg** core functions.

The main functions of **text2sdg** are `detect_sdg` and `detect_sdg_systems`, which implement the ensemble model approach (Wulff et al., 2023) and the implemented labeling systems, respectively, to identify SDGs in texts. The texts are provided to these functions via the `text` argument as either a character vector or an object of class `"tCorpus"` from **corpustools**. All other arguments are optional. By default, the `detect_sdg_systems` function runs only the Aurora, Auckland, Elsevier, and SIRIS systems, but the set systems can be extended to all six systems using the `system` argument. The functions further allow customization of the set of SDGs using the `sdgs` argument and return a `tibble` with one row per hit that has the following columns (and types) (italic column names only present in the tibble returned by `detect_sdg_systems`):

- document (factor) - index of element in the character vector or corpus supply for text
- sdg (character) - labels indicating the matched SDGs
- system (character) - the query or ensemble system that produced the match
- *query_id* (integer) - identifier of query in the query system
- *features* (character) - words in the document that were matched by the query
- hit (numeric) - running index of matches for each system

Further details on the `detect_sdg` and `detect_sdg_systems` functions and their output will be presented in the next section.

The `detect_any` function implements the same functionality as `detect_sdg_systems`, but permits the user to specify customized or self-defined queries. These queries are specified via the `queries` argument and must follow the syntax of the **corpustools** package (see Practical Considerations section for more details).

To support the interpretation of SDG labels generated by `detect_sdg`, `detect_sdg_systems` and `detect_any`, **text2sdg** further provides the `plot_sdg` and `crosstab_sdg` functions. The `plot_sdg` function visualizes the distribution of SDG labels identified in documents by means of a customizable barplot showing SDG frequencies for the different labeling systems. The `crosstab_sdg` function helps reveal patterns of label co-occurrences either across SDGs or systems, which can be controlled using the `compare` argument.

4 Demonstrating the functionality of text2sdg

To showcase the functionalities of the **text2sdg** package we analyze the publicly available p3 dataset of the Swiss National Science Foundation (SNSF) that lists research projects funded by the SNSF. In addition to demonstrating **text2sdg**, the case study will permit us to discuss practical issues concerning the labeling of SDGs, including relevant differences between labeling systems. The data to reproduce the analyses presented below can be found at `https://doi.org/10.5281/zenodo.11060662` (Meier, 2024).

4.1 Preparing the SNSF projects data

The SNSF projects data was downloaded from https://data.snf.ch/datasets. As of March 2022, the p3 database included information on 81,237 research projects. From the data, we removed 54,288 projects where the abstract was absent or not written in English. This left us with a total of 26,949 projects. To ready this data for analysis, we read it using the `readr` function of the **readr** package (Wickham et al., 2021), producing a `tibble` named `projects`. A reduced version of this `tibble` is included in the **text2sdg** package and available through the `projects` object after **text2sdg** has been loaded.

4.2 Using `detect_sdg` and `detect_sdg_systems` to detect SDGs

To label the abstracts in `projects` using `detect_sdg`, we only have to supply the character vector that includes the abstracts to the `text` argument of the `detect_sdg` function. In addition the example below makes use of the `synthetic` argument to implement the `"equal"` (default) and `"triple"` version of the ensemble model. As a result, two versions of the ensemble model are run that were trained on an equal amount of synthetic (non-SDG related) and expert-labeled data and three times the amount of synthetic than labeled data, respectively. A larger amount of synthetic data in training lowers the false-positive rate, but also compromises accuracy (cf. Wulff et al., 2023, for more details).

```
# detect SDGs
> sdgs_ensemble <- detect_sdg(text = projects,
+                             synthetic = c("equal","triple"))
Running systems
Obtaining text lengths
Building features
Running ensemble

> head(sdgs_ensemble)
# A tibble: 6 × 4
  document sdg    system            hit
  <fct>    <chr>  <chr>           <int>
1 22       SDG-06 Ensemble equal   2539
2 39       SDG-03 Ensemble equal    498
3 39       SDG-07 Ensemble equal   2953
4 39       SDG-08 Ensemble equal   4080
5 41       SDG-13 Ensemble equal   5690
6 41       SDG-13 Ensemble triple  3684
```

The first two columns of the `tibble` returned by `detect_sdg` show the document and SDGs identified by the model. Further columns show the system producing the hit and a running hit index for a given system. As the predictions of the six individual labeling systems are used as input for the ensemble models, they will be computed in the background. The user can access these predictions by calling `attr(sdgs_ensemble,"system_hits")`. Alternatively, the user can use the `detect_sdg_systems` function, which provides additional options for customization.

As with the `detect_sdg` function, the `detect_sdg_systems` function requires a character vector as input to the `text` argument. In addition, the example below specifies two optional arguments. First, to indicate that all six systems should be run, rather than the default of only Aurora, Auckland, Elsevier, and SIRIS, we supply a character vector of all six systems' names to the `systems` argument. Second, we explicitly set the `output` argument to "`features`", which in contrast to `output =` "`documents`" delivers more detailed information about which keywords that triggered the SDG labels.

```
# detect SDGs
> sdgs <- detect_sdg_systems(text = projects,
+                            systems = c("Aurora", "Elsevier", "Auckland", "SIRIS", "SDSN", "SDGO"),
+                            output = "features")
Running Aurora
Running Elsevier
Running Auckland
Running SIRIS
Running SDSN
Running SDGO
```

```
> head(sdgs)
# A tibble: 6 × 6
  document sdg    system query_id features      hit
  <fct>    <chr>  <chr>     <dbl> <chr>       <int>
1 1        SDG-01 SDSN        392 sustainable     4
2 1        SDG-02 SDSN        376 maize           3
3 1        SDG-02 SDSN        629 sustainable     8
4 1        SDG-08 SDGO       3968 work            1
5 1        SDG-08 SDSN        812 work           11
6 1        SDG-09 SDSN        483 research        6
```

The above `tibble` produced by **text2sdg** contains for every combination of document, SDG, system, and query (columns 1 to 4), the query feature (keyword) that triggered the label (column 5), and a hit index for a given system (column 6). The first row of the `tibble` thus shows that the query 392 within SDSN labeled document number 1 with SDG-01, because the document included the feature *sustainable*, and that this was the fourth hit produced by the SDSN system. It is important to note that, in other cases, multiple features of a query might be matched, which will result in multiple rows per combination of document, SDG, system, and query. This can be avoided by setting the `output` argument to "`documents`", in which case all features' hits of such combinations will be grouped into a single row.

4.3 Analyzing the SDG labels

To visualize the distribution of SDG labels across SDGs and systems in the `sdgs tibble`, we apply the `plot_sdg` function. By default, `plot_sdg` shows a barplot of the number of documents labeled by each of the SDGs, with the frequencies associated with the different systems stacked on top of each other. The function counts a maximum of one hit per document-system-SDG combination. Duplicate combinations resulting from hits by multiple queries or keywords in queries will be suppressed by default and the function returns a message reporting the number of cases affected.

```
> plot_sdg(sdgs)
139048 duplicate hits removed. Set remove_duplicates = FALSE to retain duplicates.
```

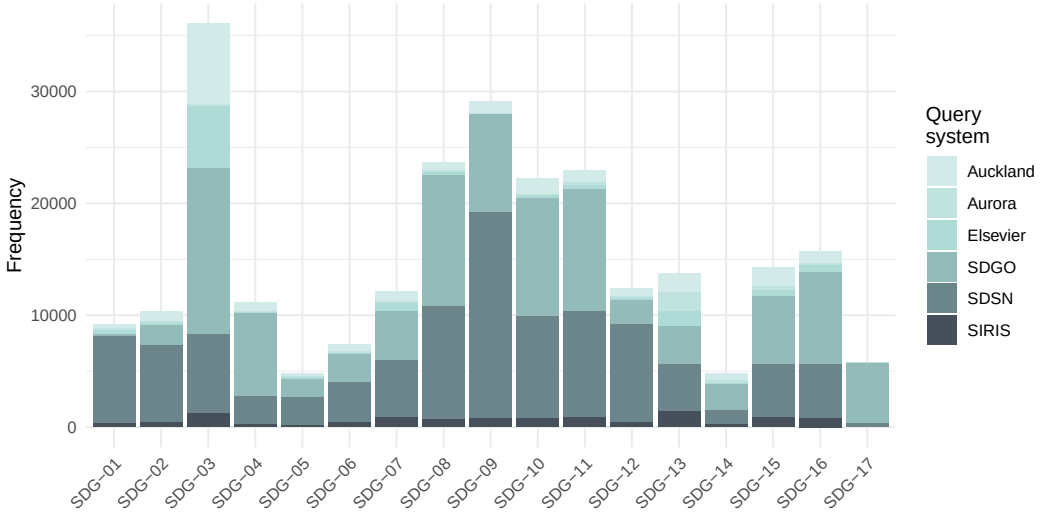

Figure 1: Default plot of distribution of detected SDGs.

The plot produced by `plot_sdg` (Figure 1) shows considerable differences in the frequency of different SDGs, with SDGs 3 ("Good Health and Well-Being") and 9 ("Industry, Innovation And Infrastructure") being most frequent and SDGs 5 ("Gender Equality") and 14 ("Life Below Water") being least frequent. Furthermore, there are substantial differences in the number of labels produced by different systems, with SDSN and SDGO having produced many more labels than the other three systems.

To customize the visualization of SDG frequencies, the `plot_sdg` function provides several additional arguments. For instance, by setting `sdg_titles` to `TRUE`, the SDG titles will be added to the annotation of the plot. Other arguments are `normalize` to show probabilities instead of frequencies, `color` to change the filling of bars, and `remove_duplicates` to eliminate duplicate document-system-SDG combinations. Furthermore, as `plot_sdg` is built on **ggplot2** (Wickham, 2016), the function can easily be extended by functions from the **ggplot2** universe. To illustrate these points, the code below generates a plot (Figure 2) that includes SDG titles and separates the results of the different SDG systems using facets.

```
> plot_sdg(sdgs,
+          sdg_titles = TRUE) +
+   ggplot2::facet_wrap(~system, ncol= 1, scales = "free_y")
139048 duplicate hits removed. Set remove_duplicates = FALSE to retain duplicates.
```

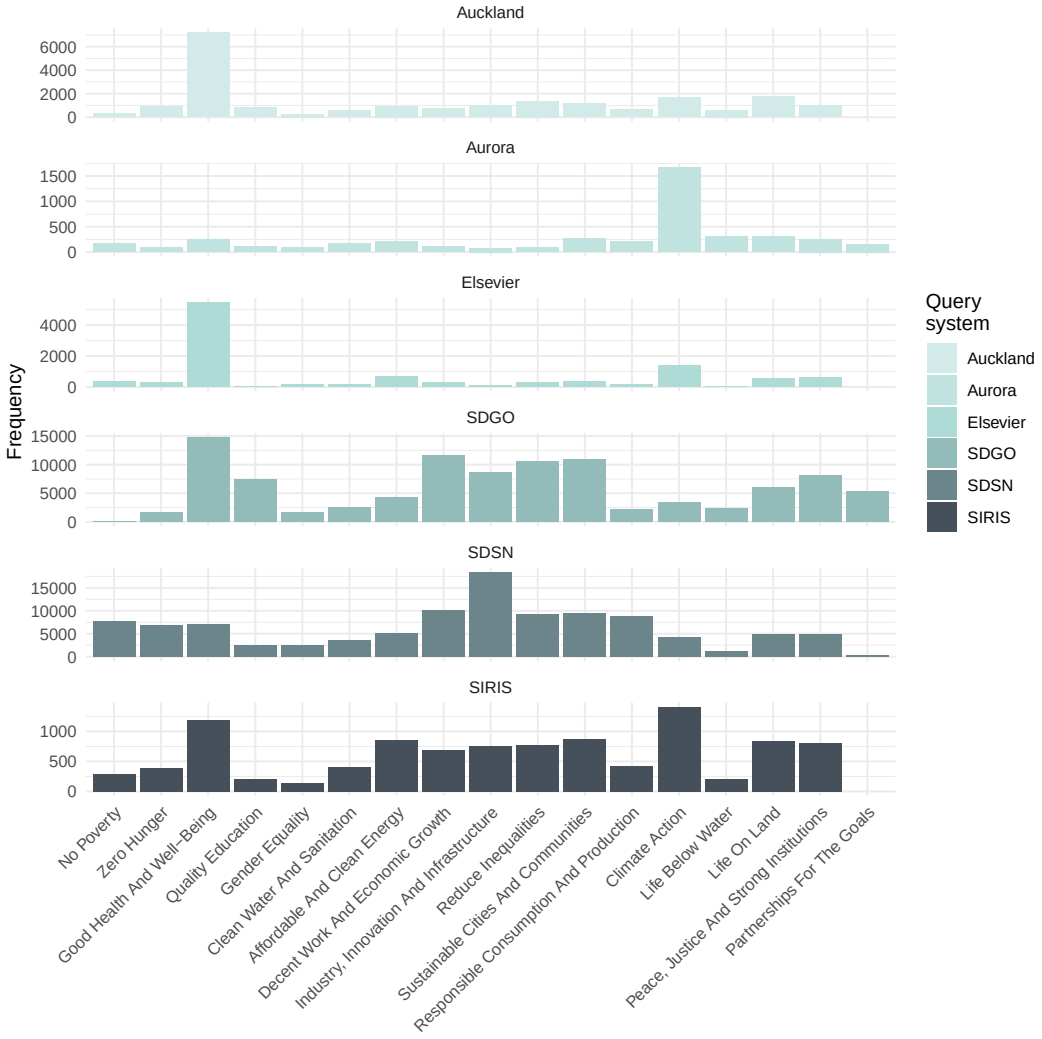

Figure 2: Distribution of detected SDGs facetted by system.

The separation of systems better illustrates the results of systems that produce fewer hits and helps compare the results across systems. This reveals, for instance, that in the Elsevier system SDG 3 ("Good Health and Well-Being") was most prominent, whereas in the Aurora system this was SDG 13 ("Climate Action"). These results highlight that the different labeling systems do not necessarily agree concerning the assignment of SDGs to documents.

To quantify the commonalities and differences between labeling systems, **text2sdg** provides the `crosstab_sdg` function. The function evaluates the level of alignment across either systems (the default) or SDGs by calculating ϕ coefficients between the vectors of labels. We supply the `hits` argument of the function with the `sdgs tibble` containing the labels produced by `detect_sdg`. Note that the function only considers distinct combinations of documents, systems and SDGs, irrespective of whether the `detect_sdg` function was run using `output = "documents"` or `output = "features"`.

```
> crosstab_sdg(sdgs)
          Auckland    Aurora  Elsevier      SDGO      SDSN     SIRIS
Auckland 1.0000000 0.3345247 0.6676524 0.3314806 0.2896650 0.4115387
Aurora   0.3345247 1.0000000 0.3256877 0.1614586 0.1569791 0.3703457
Elsevier 0.6676524 0.3256877 1.0000000 0.2642918 0.2192051 0.3538272
SDGO     0.3314806 0.1614586 0.2642918 1.0000000 0.3722997 0.2244774
SDSN     0.2896650 0.1569791 0.2192051 0.3722997 1.0000000 0.2330684
SIRIS    0.4115387 0.3703457 0.3538272 0.2244774 0.2330684 1.0000000
```

The output of `crosstab_sdg()` for the SNSF projects reveals two noteworthy insights. First, the correspondence between the labels of different systems is rather small, as indicated by ϕ coefficients that are mostly smaller than 0.4. Second, there are two groups of systems that are more similar to one another. On the one hand, Elsevier, Auckland, Aurora, and SIRIS, and, on the other hand, SDGO and SDSN. These groups correspond to differences in query operators, with the former four including AND operators in their queries, whereas the latter two do not. `crosstab_sdg()` can also be called with the output from the ensemble models.

```
> crosstab_sdg(sdgs_ensemble)
                Ensemble equal Ensemble triple
Ensemble equal       1.0000000       0.8127837
Ensemble triple      0.8127837       1.0000000
```

It can further be informative to analyze the correlations between SDGs. To do this, we set the `compare` argument in `crosstab_sdg()` to `"sdgs"`. The output below shows the result for the first six SDGs by setting `sdgs = 1:6`. It can be seen that certain pairs of SDGs—in particular, SDG-01 and SDG-02—co-occur more frequently. These results may provide insights into the co-occurrence structure of SDGs in the data at hand. However, these results can also highlight the importance of considering similarities between queries targeting different SDGs.

```
> crosstab_sdg(sdgs, compare = "sdgs", sdgs = 1:6)
           SDG-01     SDG-02     SDG-03     SDG-04     SDG-05     SDG-06
SDG-01 1.00000000 0.47455139 0.04811778 0.07928418 0.14252372 0.16622948
SDG-02 0.47455139 1.00000000 0.10611662 0.06751253 0.09338952 0.17504027
SDG-03 0.04811778 0.10611662 1.00000000 0.18092227 0.10936179 0.04882173
SDG-04 0.07928418 0.06751253 0.18092227 1.00000000 0.11791600 0.07887042
SDG-05 0.14252372 0.09338952 0.10936179 0.11791600 1.00000000 0.04603253
SDG-06 0.16622948 0.17504027 0.04882173 0.07887042 0.04603253 1.00000000
```

5 Practical considerations

5.1 Specifying user-defined labeling systems

The query systems implemented in **text2sdg** represent important efforts to systematize the monitoring of SDGs from text. Nevertheless, these efforts are still relatively young and validations of the systems are largely missing, creating a need for continued development. **text2sdg** supports the further development of new SDG labeling systems by providing the `detect_any` function. In this section, we provide additional detail on using this feature of **text2sdg**.

The `detect_any` function also uses **corpustools** as the back-end. This implies that new queries must be specified to match the syntax of **corpustools**. The syntax supports standard Boolean operators (AND, OR, and NOT), wildcard operators, and proximity search. Boolean operators control how different keywords are combined in a query. For instance, the query "marine OR fisheries" matches text that contains either of these two words whereas the query "marine AND fisheries" only matches text that contains both words. Corpustools also allows to specify common query wildcard operators [1]. The wildcard operators ? and $*$ allow the specification of variable word parts. For instance, the question mark operator ? matches one unknown character or no character at all, e.g., "?ish" would match "fish", "dish", or "ish". The asterisk operator $*$, by contrast, matches any number of unknown characters, e.g., "*ish" would match "fish" but also "Swedish". Both wildcards can be used at the start, within or end

[1] Note that the meaning of these wildcards differs from regex wildcards.

of a term. Proximity search extends a Boolean AND, by requiring that two keywords have no more than defined distances to one another. For instance, "climate change"∼3 specifies matches in which "climate" and "change" both occur no more than three words apart. A complete description of the **corpustools** syntax is presented in the **corpustools** vignette and documentation.

To supply a user-defined labeling system to `detect_any`, the queries must be placed in a `data.frame` or `tibble` that additionally includes a column specifying the labeling system's name and a column of SDG labels corresponding to the queries.

- system (character) - name of the labeling systems.
- queries (character) - user-defined queries.
- sdg (integer) - SDGs labels assigned by queries.

The example below illustrates the application of a user-defined labeling system using `detect_any`. First, a `tibble` is defined that includes three rows, one for each of three different queries stored in the `query` column. The system is called `"my_example_system"` in the `system` column and each of the queries is assigned SDG-14 in the `sdg` column. Note that specification of the labeling system need not be made in R, but can easily be outsourced to a spreadsheet that is then processed into a `tibble`. Second, the system is supplied to the `system` argument of the `detect_any` function, along with the texts (here, the SNSF abstracts). The output is analogous to the output of the `detect_sdg_systems` function (for brevity, we only show the first three lines of the output).

```
> # definition of query set
> my_example_system <- tibble::tibble(system = "my_example_system",
+                              query = c("marine AND fisheries",
+                                        "('marine fisheries') AND sea",
+                                        "?ish"),
+                              sdg = c(14,14,14))
> detect_any(text = projects,
+            system = my_example_system)
# A tibble: 591 × 6
   document sdg    system            query_id features   hit
   <fct>    <chr>  <chr>                <dbl> <chr>    <int>
 1 6        SDG-14 my_example_system        3 wish       122
 2 134      SDG-14 my_example_system        3 wish        18
 3 241      SDG-14 my_example_system        3 fish        59
```

5.2 Applying text2sdg to non-English data

The queries of the labeling systems implemented by **text2sdg** are in English, implying that texts in other languages must first be translated to English. We assessed feasibility and whether translation affects the reliability of SDG labels by making use of back translation with one language we are most familiar with (German). To this end, we first translated 1,500 randomly selected SNSF project abstracts from English to German and from German to English and then compared the labels of the original English and back-translated English abstracts. We carried out the translation using the DeepL translation engine (www.deepl.com/translator).

Table 3 shows the results of this analysis. Overall, the correlations as measured by the *phi*-coefficient are very high. The systems showed correlations above or equal to 0.88, with Elsevier and Auckland showing the highest value of 0.93. Considering that our analysis involves not only one, but two translation steps—from German to English and back—these results suggest that **text2sdg** can be applied to non-English text, such as German, with very high accuracy. One should note, however, that the quality of translation may vary across languages and translation engines so additional work is needed to compare performance across different languages.

Aurora	Elsevier	Auckland	SIRIS	SDSN	SDGO
0.91	0.93	0.93	0.88	0.91	0.91

Table 3: *phi*-coefficient between the labels for the original English text and the labels for the back-translated (English-German-English) English text

5.3 Estimating the runtime of text2sdg

The analysis of text data can be computationally intense. To provide some guidance on the expected runtime of **text2sdg** for data with different numbers of documents and different document lengths, we carried out several experiments. For this purpose, we first simulated documents by concatenating 10, 100, 1,000, or 10,000 words drawn randomly according to word frequencies in Wikipedia and combined 1, 10, 100, or 1,000 thus-generated documents into simulated data sets. Then we evaluated the runtime of **text2sdg** separately by system for the simulated data sets.

Figure 3 shows the average runtime in seconds across 7,000 repetitions of each combination of document length and number of documents for each of the labeling systems. The results highlight noteworthy points. First, runtime is primarily a function of the number of words, irrespective of how words are distributed across documents. Second, the runtime per words decreases as the number of words increases, which is due to a constant overhead associated with optimizing the labeling systems' queries. Third, there are considerable differences in the runtime between systems, which is, in part, due to the functions' overhead and, in part, due to differences in number and complexity of queries. The fastest system is Elsevier, processing 10 million words in roughly one minute; the slowest system is SIRIS, processing 10 million words in about 40 minutes. Overall, these experiments highlight that **text2sdg** can efficiently process large amounts of text, but also that some care should be exercised when dealing with extremely large or many texts. In such cases, it may be advisable to rely on more efficient labeling systems, such as Elsevier or SDSN.

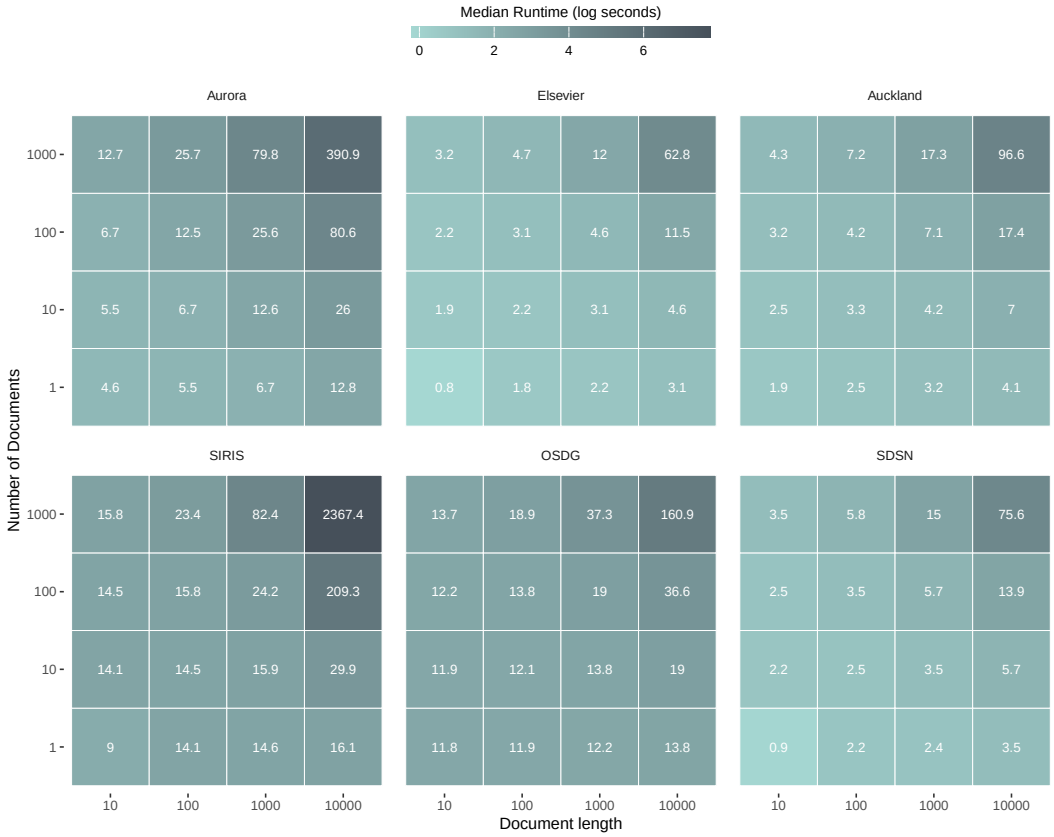

Figure 3: Median runtime as a function of number of documents and document length using 6 different query systems. Each cell reflects the average runtime of 7,000 runs with numbers reflecting the median runtime in seconds and color reflecting the logarithm of the median runtime in seconds.

6 Other approaches to detecting SDGs in text

There are a number of other approaches to detecting SDGs in text. First, there are approaches outside the R ecosystem. One such tool is the European Union's SDG Mapper (`https://knowsdgs.jrc.ec.europa.eu/sdgmapper`) that produces an analysis of SDGs per document using an online interface in which registered users can upload single documents. Another prominent example is the OSDG tool developed by the SDG Ai Lab of the United Nations in collaboration with private partners. It can

detect SDGs in text that is provided through the OSDG website (https://osdg.ai/) or, if granted access, through an API. The OSDG tool builds on the SDG Ontology (SDGO) that is also implemented in **text2sdg**. OSDG additionally leverages a machine learning tool that was trained on expert-labeled data to make the final predictions (Pukelis et al., 2022). One advantage of OSDG relative to **text2sdg** is that it allows to detect SDGs in 15 different languages. This is done by using translation of the input text into English before passing it through the OSDG workflow. While this is convenient to the user, the same outcome can be achieved with our package by making use of translation models through, for example the **deeplr** R package. As our proof-of-concept above has shown, **text2sdg** can be used with non-English text (e.g., German) with very high accuracy by using such an approach.

Second, there are currently, to our knowledge, two other R packages aimed at providing methods for the automated detection of SDGs in text. The **SDGdetector** package is based on a custom query system that was generated by pooling several existing query systems and manual adaptions. The resulting labeling system permits finer-grained predictions on the level of SDG targets [2]. However, the method is computationally taxing and limited to texts that are shorter than 750 characters or approximately 150 words. The **SDGmapR** package builds on publicly available SDG keywords that are assigned weights that indicate the degree to which a keyword reflects a given SDG. The package computes SDG weights for each text by adding up the weights of the keywords that were found in the text. The larger this weight, the larger should be the likelihood that the text is related to a specified SDG. The advantage of this approach is that it permits customization of the decision boundary (i.e., the weight needed to count a text as SDG related). However, the package does not give the user a binary decision regarding whether a text relates to a given SDG. None of the two packages offers an ensemble model that can be used to categorize the presence of SDGs as is the case with **text2sdg**.

7 Discussion

The **text2sdg** package offers an open and easily accessible way of detecting SDGs in text using both individual query systems, a state-of-the-art ensemble model that combines queries from extant systems (Wulff et al., 2023), as well as custom-made queries.

While our package implements several query-based methods to detect SDGs in text as well as a state-of-the-art ensemble model, the field of detecting SDGs in text is rapidly evolving. Our aim is to continuously update **text2sdg** as new open source methods of detecting SDGs in text are released. Bundling many systems in a coherent API is not only convenient for users, but also helps catalyze development of new and hopefully more accurate methods by making it easy to compare the performance of the different systems. We deliberately incorporated functions that allow users to implement and test their own query systems to facilitate this process. We also encourage others to contribute to **text2sdg** by adding new systems or by expanding the existing functionalities to analyse the output of the systems.

Indeed, although the systems implemented by **text2sdg** have been shown to achieve high accuracy (Wulff et al., 2023), it is important to stress that these systems must be further developed to increase their accuracy for a greater number of document types. Two approaches can help in achieving this. First, unsupervised methods such as topic models (Grün and Hornik, 2011) or semantic network analysis (Siew et al., 2019) can help in identifying novel linguistic patterns for the detection of SDGs. One should note, however, that unsupervised methods are no replacement for top-down, rule-based methods as implemented by **text2sdg**, because of the strong requirement to compare results across data sets, analyses, and time, which require a clear set of benchmarks that are not simply data-driven. Second, recent transformer based models (Reimers and Gurevych, 2019) could be leveraged to learn more complex relationships between specific linguistic patterns and SDGs. However, the field will have to work towards producing more balanced training data before the full potential of these approaches can be exploited. Moreover, one should note that transformer models are computationally expensive and often limited to short text due to architecture constraints (Ding et al., 2020). Whether such developments will emerge and can be ultimately integrated into **text2sdg** or will represent alternative approaches remains an open question.

8 Conclusion

In this article, we introduced a new R package, **text2sdg**, designed to help identify SDGs from text. The package promises to help detect SDGs in text sources using different existing or custom-made

[2] Each SDG has several targets that are operationalized with indicators (SDG/targets/indicators). For example the first target of SDG 1 reads as follows: "By 2030, eradicate extreme poverty for all people everywhere, currently measured as people living on less than $1.25 a day".

labeling systems as well as a high-performance ensemble model that builds on these labeling systems. Our case study and additional analyses suggest that the approach can handle both sources in English as well as translations, allows user-friendly use of novel queries, and provides reasonably efficient performance for analysing large corpora.

References

N. Bautista. Sdg ontology. 2019. URL https://doi.org/10.6084/m9.figshare.11106113.v1. [p84]

M. Ding, C. Zhou, H. Yang, and J. Tang. Cogltx: Applying bert to long texts. *Advances in Neural Information Processing Systems*, 33:12792–12804, 2020. [p93]

N. Duran-Silva, E. Fuster, F. A. Massucci, and A. Quinquillà. A controlled vocabulary defining the semantic perimeter of Sustainable Development Goals, Dec. 2019. URL https://doi.org/10.5281/zenodo.3567769. [p84]

B. Grün and K. Hornik. topicmodels: An r package for fitting topic models. *Journal of statistical software*, 40:1–30, 2011. doi: https://doi.org/10.18637/jss.v040.i13. [p93]

B. Jayabalasingham, R. Boverhof, K. Agnew, and L. Klein. Identifying research supporting the united nations sustainable development goals. *Mendeley Data*, 1, 2019. URL https://doi.org/10.17632/87txkw7khs.1. [p83, 84]

D. S. Meier. Descriptions of snsf-funded research projects, Apr. 2024. URL https://doi.org/10.5281/zenodo.11060662. [p86]

L. Pukelis, N. B. Puig, M. Skrynik, and V. Stanciauskas. Osdg–open-source approach to classify text data by un sustainable development goals (sdgs). *arXiv preprint arXiv:2005.14569*, 2020. doi: https://doi.org/10.48550/arXiv.2005.14569. [p85]

L. Pukelis, N. Bautista-Puig, G. Statulevičiūtė, V. Stančiauskas, G. Dikmener, and D. Akylbekova. Osdg 2.0: a multilingual tool for classifying text data by un sustainable development goals (sdgs), 2022. URL https://arxiv.org/abs/2211.11252. [p93]

N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019. doi: https://doi.org/10.48550/arXiv.1908.10084. [p93]

C. S. Siew, D. U. Wulff, N. M. Beckage, and Y. N. Kenett. Cognitive network science: A review of research on cognition through the lens of network representations, processes, and dynamics. *Complexity*, 2019, 2019. doi: https://doi.org/10.1155/2019/2108423. [p93]

Sustainable Development Solutions Network (SDSN). Compiled list of sdg keywords, 2021. URL https://ap-unsdsn.org/regional-initiatives/universities-sdgs/. [p84]

UN. *The Sustainable Development Goals Report 2022*. United Nations, 2022. [p83]

M. Vanderfeesten, R. Otten, and E. Spielberg. Search Queries for "Mapping Research Output to the Sustainable Development Goals (SDGs)" v5.0, July 2020a. URL https://doi.org/10.5281/zenodo.3817445. [p84]

M. Vanderfeesten, E. Spielberg, and Y. Gunes. Survey data of "Mapping Research Output to the Sustainable Development Goals (SDGs)", May 2020b. URL https://doi.org/10.5281/zenodo.3813230. [p84]

W. Wang, W. Kang, and J. Mu. Mapping research to the sustainable development goals (sdgs). 2023. URL https://doi.org/10.21203/rs.3.rs-2544385/v1. [p84]

K. Welbers and W. van Atteveldt. *corpustools: Managing, Querying and Analyzing Tokenized Text*, 2021. URL https://CRAN.R-project.org/package=corpustools. R package version 0.4.8. [p86]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org. [p89]

H. Wickham, J. Hester, and J. Bryan. *readr: Read Rectangular Text Data*, 2021. URL https://CRAN.R-project.org/package=readr. R package version 2.1.1. [p87]

D. U. Wulff, D. S. Meier, and R. Mata. Using novel data and ensemble models to improve automated labeling of sustainable development goals. *arXiv preprint arXiv:2301.11353*, 2023. [p85, 86, 87, 93]

Dominik S. Meier
University of Basel
Steinengraben 22 4051 Basel
Switzerland
(ORCID: 0000-0002-3999-1388)
dominik.meier@unibas.ch

Rui Mata
University of Basel
Missionsstrasse 60-62 4055 Basel
Switzerland
(ORCID: 0000-0002-1679-906X)
rui.mata@unibas.ch

Dirk U. Wulff
University of Basel
Missionsstrasse 60-62 4055 Basel
Switzerland
(ORCID: 0000-0002-4008-8022)
dirk.wulff@unibas.ch