TEXTCLASSIFY ALGORITHM AND ITS USE

GENERAL INTRODUCTION TO TEXTCLASSIFY

The "TextClassify" algorithm is a classification algorithm where texts get classified into different well-defined categories. The algorithm works on small extracts of text called "TextSnippet". These typically correspond to a paragraph containing a specific meaning.[1] TextClassify can be trained to recognize statistical patterns of words based on manual annotations or "Links", called "GroundTruth", within a training sample. This manual annotation/tagging of GroundTruths is done by human experts, that define which Textsnippets are within the definition of a specific category (GroundTruth=true also called Signal) and which are not (GroundTruth=false also called Background). This is also called supervised machine learning. The training process optimizes the TextClassify performance and helps improve the annotation by tagging outlier TextSnippets to be further scrutinized.

From the training sample TextClassify constructs the probability, "Purity", that any TextSnippet is annotated within a given category. After training, Purity can be evaluated on huge amounts of data. TextSnippets are selected when Purity within a category is above a certain value, which is a simple cut-based approach. The default cut off is normally Purity>50%, but this value can be altered.

The performance number is given as the number of annotated TextSnippets that the algorithm recognizes to be linked to a specific category. A performance of 50% means that the average purity of the GroundTruths=true within a specific category is 50%. The performance measure is also known as the number of "MachineLearnedLinks". Typically, a performance number >50% indicates a useful analysis, but this cut always depends on the analysis in question.

An important aspect of TextClassify is its speed. This enables simultaneous training on hundreds of different Categories and using training samples up to order of 1 million TextSnippets. All this happens on a normal PC within a few minutes.

THE MODEL

TextClassify assumes that the meaning of a TextSnippet can be boiled down to the presence of a maximum of three components and these are constrained to individual (or simple combinations of) words. The precise position inside the text is considered irrelevant. In a simple interpretation, the 3 components, "Types", behave like a linguistic Verb-Subject-Object (VSO) model. The presence of the 3 Types (V S O) is described by their strength in linking, "Levels". The values of the Level for each Type are restricted to run from 0 to 3 (3 being a strong link), and the combined output is simply the maximum Level for each Type.

To maximize both the performance and robustness of TextClassify, different inflections

of the same (stem) word are treated equally. These stem words are treated as the basic building blocks for the pattern recognition. Regular expressions are built into TextClassify by combining words occurring close to each other into what we call “WordCombine”. On top of this, ambiguous interpretations are allowed, e.g., “malaria” can be interpreted as both “Malaria” and a synonym “IsDisease”. These two simple building blocks can be used to expand the dictionary TextClassify works with. TextClassify will try any potential expansions, order them according to their performance, and output prioritized lists, so the user can manually accept any of the proposed expansions.

RUNNING TEXTCLASSIFY

TextClassify runs using standardized input and output files. To run and train the algorithm, the following input files are needed:

- A file that contains TextSnippets and one or several columns with the defined categories containing GroundTruth.
- A Stem-dictionary which contains all usable words in terms of search patterns and how they are handled. This constitutes a programmable dictionary, which can be changed according to need.
- An expert-dictionary which contains the chosen expert words relevant for the defined categories, “WordExpert”, with Type (VSO) and Level (0-3) for each Category.

When TextClassify runs, all possible improvements of each Expert word are tested (optimized), for all WordExpert, and for all categories. Moreover, all possible extensions of TextClassifyExpert are tested. All changes (with high performance) are applied after approval by the user. The user can (manually) add new words in the Expert Dictionary and also change the Stem-dictionary. Improvements in performance is measured as an increase in the number of MachineLinks.

In practice, the user starts by classifying (making GroundTruth) on several TextSnippets. Then, TextClassify is run. TextClassify spits out 1) a wide range of files with relevant information to optimize the results and 2) the actual results of the evaluation of the algorithm in terms of purity values within each defined category.

The user will typically sort the TextSnippets by Purity, compare with GroundTruth, and perform the following:

1. Examine false positives (high Purity without GroundTruth). These normally occur when the user has overlooked the text snippet (the missing GroundTruth is inserted).
2. Investigate false negatives (low Purity with GroundTruth). These are either due to the user’s tagging a text snippet as a GroundTruth for no reason (the erroneous GroundTruth is removed) or that a relevant component of the TextSnippet is not represented in the Expert Dictionary. The reason why TextClassify provides a low Purity is available to the user, who can then use the information to identify potential missing WordExpert(WordCombine), which is then inserted into the Expert Dictionary.

3. When 1 + 2) is executed, TextClassify can be rerun run with improved performance as a result.
4. When the learning curve is no longer steep, the process ends. A typical scenario includes scrutinizing TextSnippets with Purity around 50%. If these make sense the learning has converged, otherwise these TextSnippets should be used to potentially improve the algorithm.
5. When the training is optimized, it can be used to evaluate enormous amounts of other texts very fast.

With a cut of 50% false positives are defined as Purity → 50 & GT ≠ True, where false negatives are defined as Purity < 50 & GT = True. See table 1.

Table 1: Confusion matrix

		Actual Expert tagging	
		Positive GT = True	**Negative** GT ≠ True
Predicted Classification TC8	**Positive** Purity >= 50	**True Positive** Purity → 50 & GT = True	**False Positive** Purity → 50 & GT ≠ True
	Negative Purity < 50	**False Negative** Purity < 50 & GT = True	**True Negative** Purity < 50 & GT ≠ True

TextClassify runs on one "Category" of meaning at a time. When the training is optimized, the algorithm can be used without the training sample of text snippets. This is the so-called StandAlone version of the algorithm.

LIMITATIONS

Using a machine learning algorithm like TextClassify makes one able to classify a large amount of text in a very short time. It does, however, comes with the likelihood that some snippets will be misclassified. Misclassification means that some snippets will be linked to a category even though they do not belong in that category and that some snippets will not be linked to a category even though this would be correct – this is similar to what is described as false positive and false negative in the previous section but without the possibility of verification based on a tagged GroundTruth. The likelihood of misclassification of TextSnippets is mostly due to the limitations of working with machine learning technology and to the quality of the training data (the GroundTruths). For the training part of TextClassify to be optimal it requires both the full meaning to be present in the TextSnippets (no context) and not too many meanings to be present (combinatorial noise). Most limitations are related to fuzzy and rare categories.

When working with supervised machine learning one should also be aware of the fact, that the tagging of GroundTruths is done by humans. Humans can disagree on the meaning of a text, humans can make mistakes in tagging, and humans can be biased. That human experts can have different interpretations of a text should make the algorithm less biased towards one understanding. However, this difference in tagging can also lead the algorithm to be less precise.

Also, TextClassify is not able to understand nuances or implicit meanings of a text. It can only classify text based on information obtained during the training. Therefore, TextClassify cannot determine whether a text snippet contains positive or negative statements about a certain topic and is not able to categorise a text based on an implicit understanding of that text. TextClassify will understand a text literally as it is stated.

PERFORMANCE

To evaluate how well TextClassify is performing for each category, meaning how accurate TextClassify classifies TextSnippets within a certain category, different performance measures can be applied. The performance of TextClassify is evaluated both with and without a cut-based approach, where the cut-based approach is the one we normally use. The evaluation is based on the F1-score (Best F1) and the Integrated Efficiency * Purity (IntE*P) for each category. The performance derived using the cut-based approach indicates the performance with the default cut of 50 % while the performance measured without the cut is a more general way of evaluating the performance.

The F1-score (Best F1) is a common way to evaluate the accuracy of a machine learning algorithm. It combines the purity (also known as precision) and the efficiency (also known as recall) scores of a model.[2] It provides a way to evaluate the model's ability to correctly classify positive instances while minimizing both false positives and false negatives. The F1-score[3] more or less corresponds to the squared root of the integrated efficiency*purity (IntE*P), which is a more precise measure of performance. The integrated efficiency*purity considers both the ability of the algorithm to identify positive instances and the accuracy of its positive predictions. IntE*P can be interpreted as the expected number of annotated TextSnippets which are later usable, also known as the number of "MachineLearnedLinks". Normally an IntE*P above 50% and/or and F1-score above 70% indicates a useful analysis, but this threshold always depends on the analysis in question. See more about the performance measures in the Appendix.

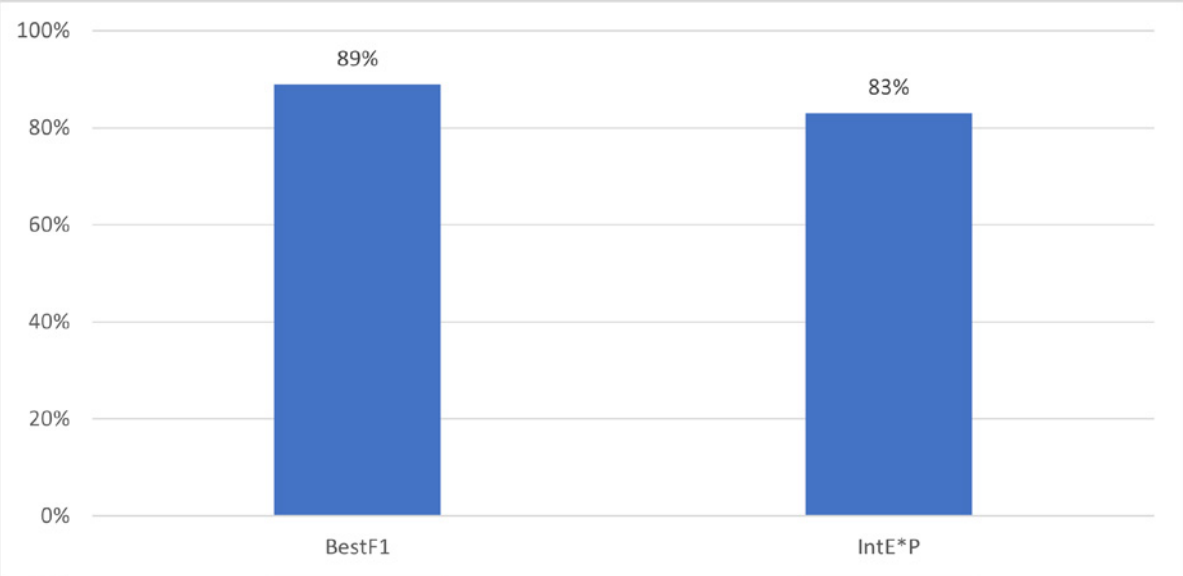
The performance of TextClassify can only be evaluated using the training data. Hence, the performance values are based on how well the algorithm classifies the training data and the expected performance when applying the algorithm therefore also depends on **the type of text** that are analysed. Using the algorithm on text that are different from the text used for training can and probably will impact the performance.[4] Evolution in language, exemplified by the emergence of terms like "MeToo" to address specific subjects, may result in instances where the algorithm fails to identify certain language patterns. This is because such terms were non-existent during the training of the algorithm.

A list of available performance measures can be accessed in the appendix where also all the performance values for each category can be found. Below is an example of how to assess the performance of the algorithm for a selected category.

Example: Performance of SDG-Target 1.2

TextClassify is among other things used to link recommendations from UN human rights monitoring bodies to several of the Sustainable Development Goals, targets, and Rights-holder groups. As an example of how to assess the performance of the algorithm, figure 1 shows the F1 value and the IntE*P for SDG-Target 1.2. Both performance measures indicate good performance as the IntE*P is above 50% and the F1 is above 70%.

Figure 1: F1 and IntE*P for Target 1.2

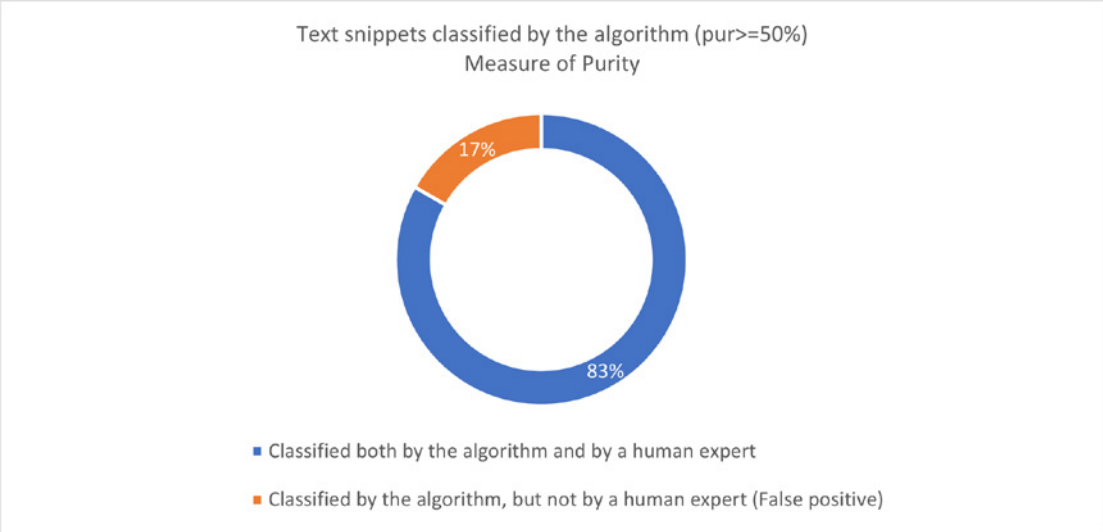


Even if a category has high performance measured by either F1 or IntE*P, it does not mean that the algorithm can classify without flaws. The measures of purity (precision) and efficiency (recall) separately give us an idea of what margin of error to expect. The purity (precision) for each category is given by the share of true positives (GT=T and Pur>=50%) out of all TextSnippets selected by the algorithm (PUR>50 %) and indicates how precise the classification is. The purity takes values from zero to one. A high purity means that when the algorithm predicts a positive result, it is usually correct.

The efficiency(recall) of each category is given by the share of true positives (GT=T and Pur>=50%) out of all TextSnippets selected by human experts (GT=T) and indicates the degree of recognition. The efficiency takes values from zero to one. A high efficiency means the model is good at identifying most of the positive cases (GT = True). Figure 2 and figure 3 shows the purity and efficiency for SDG-Target 1.2. 83 percent of the cases where the algorithm suggests a link between SDG-Target 1.2 and a recommendation, this linked has also been established by a human expert, see Figure 2. Hence, the Purity value for SDG-Target 1.2 is 83 %. This also means that in 17 % of the cases where a recommendation is linked to SDG-Target 1.2 by the algorithm,

this link has not been established by a human expert. This means that of all the recommendations that are linked to SDG-Target 1.2 by the algorithm, 17 % are False Positives.

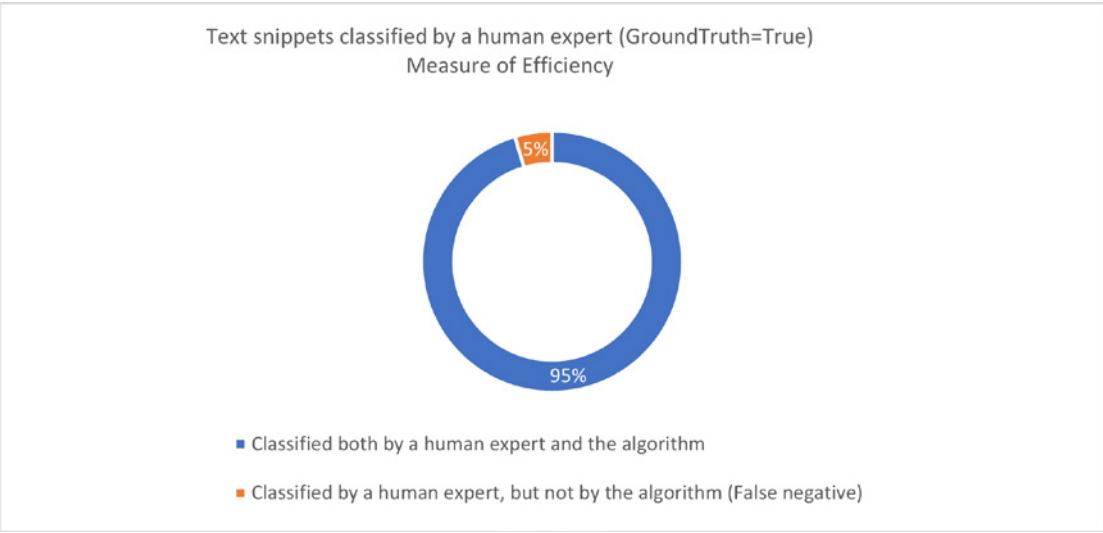
Figure 2: Text snippets selected by the Algorithm divided by whether they are also tagged by a human expert (GT=T)



Note: N=1.070 (Total number of snippets selected by the algorithm)

Figure 3 shows the efficiency of the algorithm on SDG-Target 1.2. 95 % of the cases where a human expert has linked a recommendation to SDG-Target 1.2, this link has also been created by the algorithm. Hence, the Efficiency value for SDG-Target 1.2 is 95 %. On the other hand, this also means that in 5 % of the cases where a recommendation is linked to SDG-Target 1.2 by a human expert, this link is not recognised by the algorithm. Therefore, out of all the recommendation that has been tagged as GroundTruth=true by a human expert, 5 % are False Negatives. This is also known as the False Negative Rate.

Figure 3: Text snippets selected by a human expert divided by whether they are also selected by the algorithm (Pur>=50%)



Note: N=934 (Total number of snippets tagged by human experts)

APPENDIX: EXPLANATION OF PERFORMANCE FILE

The performance file is an Excel file. The first sheet contains information about the performance measures and the second sheet contains the summarized performance information for all the categories.

OVERVIEW OF COLUMNS IN PERFORMANCE FILE (SHEET "PERFORMANCE FILE")

- **Category**: The name/code for the category, for example SDG targets, human rights themes, or rights-holder groups
- **Match**: number of text snippets that are tagged both by the expert and the algorithm, in other words, the True Positives (Pur → 50 & GT=True).
- **Total experts:** Number of text snippets that are tagged by an expert as corresponding to a category (Ground Truth = "T")
- **Total algorithm**: Number of text snippets that are tagged by the algorithm as corresponding to a category (Pur → 50)
- **False positives**: Flagged as True by the algorithm (with Pur → 50), but not confirmed as True by the expert tagging (where GT=False & Pur → 50).
- **False negatives**: Confirmed as True by the expert tagging (with GT=True), but not identified as True by the algorithm (with GT=True & Pur < 50).
- **Recall**: Measures the proportion of actual positive instances that were correctly predicted by the algorithm. In other words, it tells you what percentage of the actual positive cases the algorithm was able to correctly identify. This metric is also known as True Positive Rate, or also known as Sensitivity.

Formula	Interpretation
$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$	Range of values between **0 and 1**

- A high recall means the model is good at identifying most of the positive cases (GT = True).
- A low recall means the model is missing a significant number of positive cases.

- Precision: Gives the **percentage of the predicted positive cases (Pur → 50) that are actually correct (GT = True).**

Formula	Interpretation
$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$	Range of values between **0 and 1**

- A high precision means that when the algorithm predicts a positive result, it is usually correct. A precision of 1 would mean that all the positive predictions made by the algorithm are correct.
- A low precision means that a significant proportion of the predicted positive cases by the algorithm are incorrect.

- **1-Recall:** Quantifies the **proportion of actual positive cases (GT = True) that were missed or not detected by the algorithm (Pur < 50)**. Also known as **False Negative Rate.**

Formula	Interpretation
$1 - Recall = \frac{False\ Negatives}{GT = True}$	Range of values between **0 and 1**

A lower value is desirable. A high false rate indicates that the model is failing to identify a significant number of actual positive cases (GT=True).

- **False Positive Rate:** the **proportion** of **positive predictions made by the algorithm that were incorrect** (i.e., false positives).

Formula	Interpretation
$False\ Positive\ Rate = \frac{False\ Positives}{False\ Positives + True\ Negatives}$	Range of values between **0 and 1**

A lower value is desirable. A false positive rate of 1 would mean that every negative instance (GT ≠ True) is incorrectly predicted as positive (Pur → 50).

- **Efficiency**: same as Recall.
- **Purity**: same as Precision.
- **Efficiency*Purity**: Is equivalent as **Recall*Precision**. It provides a balanced assessment of the algorithm's performance. It considers both the ability of the algorithm to identify positive instances and the accuracy of its positive predictions. **A high Recall * Precision** indicates a model that is **both good at identifying positive cases and making accurate positive predictions**.

F1 score: Is the standard metric which combines the Precision and the Recall into a single measure of model performance. It provides a way to **evaluate the model's ability to correctly classify positive instances while minimizing both false positives and false negatives.**

Formula	Interpretation
$F1 \text{ - } score = \frac{2 \text{ x } True\ Positives}{2 \times True\ Positives + False\ Positives + False\ Negatives}$	Range of values between **0 and 1**

- A F1 score of 1 indicates perfect precision and recall, meaning that the model has no false positives or false negatives. This is the best possible score.
- A F1 score of 0 indicates that either precision or recall is 0, meaning that the model is performing very poorly.

- **Number of GT=True**: Number of text snippets marked as ground truth for a specific category by an expert.

ENDNOTES

1 The SDG version of the TextClassify automatic links recommendations from UN human rights monitoring bodies to several of the Sustainable Development Goals and targets. Here the goals and targets serve as categories and the individual recommendations serve as TextSnippets. To see more on the specific methodology behind the SDG version of TextClassify consult the homepage: https://sdgdata.humanrights.dk/en/methodology

2 Where Purity and Efficiency is given by the formulas:

$$Purity = \frac{number\ selected\ GT = True}{total\ selected\ GT = True + False} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Efficiency = \frac{number\ selected\ GT = True}{total\ GT = True} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

3

$$F1 = \frac{2}{\frac{1}{Efficiency} * \frac{2}{Purity}} = \frac{Efficiency * Purity}{Efficiency + Purity} = \frac{True\ Positive}{True\ Positive + ½\ (false\ pos. + false\ neg.)}$$

4 Using the SDG version of the TextClassify on similar text as UN recommendations will enable a more or less direct translating of the performance while using the algorithm on other type of text snippets will make it more difficult to use the performance measures.