# Online suicide prevention through optimised text classification

Bart Desmet and Véronique Hoste

*LT3, Language and Translation Technology Team, Ghent University*
*Groot-Brittanniëlaan 45, 9000 Ghent, Belgium*
*bart.desmet,veronique.hoste@ugent.be*

**Abstract**

Online communication platforms are increasingly used to express suicidal thoughts. There is considerable interest in monitoring such messages, both for population-wide and individual prevention purposes, and to inform suicide research and policy. Online information overload prohibits manual detection, which is why keyword search methods are typically used. However, these are imprecise and unable to handle implicit references or linguistic noise. As an alternative, this study investigates supervised text classification to model and detect suicidality in Dutch-language forum posts. Genetic algorithms were used to optimise models through feature selection and hyperparameter optimisation. A variety of features was found to be informative, including token and character ngram bags-of-words, presence of salient suicide-related terms and features based on LSA topic models and polarity lexicons. The results indicate that text classification is a viable and promising strategy for detecting suicide-related and alarming messages, with F-scores comparable to human annotators (93% for relevant messages, 70% for severe messages). Both types of messages can be detected with high precision and minimal noise, even on large high-skew corpora. This suggests that they would be fit for use in a real-world prevention setting.

*Keywords:* suicide prevention, social media, text classification, machine learning, feature selection, optimisation

## 1. Introduction

Suicidal behaviour is an important public health concern. Globally, an estimated one million people die by suicide each year [42], making it the sixth leading cause of death for adults aged 20 to 59 years, and the primary cause of death among teenagers [45]. Apart from successful suicides, there are ten to twenty times as many non-fatal attempts, which also have disruptive emotional and economic consequences. Suicide ideation has an even higher incidence: in a Belgian survey, suicidal thoughts were found to have affected 10% of the male and 15% of the female population between 15 and 24 years old [10].

In spite of these alarming numbers, suicide is generally considered a preventable death: regardless of a victim's stage in the suicidal process (i.e. the progressive stadia of suicidal thoughts, attempt(s) and actual suicide), there often remains ambivalence between life and death. It is a common adage in prevention discourse that suicide is a permanent solution to a temporary problem. Prevention is typically aimed at either the general population, by reducing risk factors and removing barriers to mental health access, or at people who are known or expected to have suicidal tendencies, with adequate risk assessment, medication, therapy and acute crisis support (e.g. suicide hotlines). However, these two prevention types fail to adequately reach the blind spot in between: at-risk individuals who have not yet exhibited suicidal behaviour or found their way to secondary prevention. Efforts to bridge that gap may benefit significantly from suicidality detection on social media.

The rise of the 'social' Web 2.0 has had far-reaching implications for human communication. It opened up the possibility to interact and form communities online. Inevitably, these developments have also had an impact on how people communicate about suicidal behaviour. [32] found evidence of reduced inhibition and more self-disclosure in online communication, since it can offer anonymity and a sense of control. Social media have indeed become an outlet for people contemplating suicide to share their thoughts and feelings. Such suicidal expressions can be recognized and responded to by peers, although this may

happen in an inappropriate or untimely fashion, if at all. It is therefore preferable to also have trained website administrators or suicide prevention workers monitor user-generated content, if this is not in conflict with users' preferences, safety and privacy concerns.

Given the massive volume of online content that is continually produced, manual monitoring is practically infeasible. Automatic approaches are therefore required. A search-based approach that uses keywords to locate relevant content would reduce the volume, but still presents a number of problems:

- Specific search queries may only cover a limited range of explicit suicidal expressions (e.g. *suicide* or *kill myself*). Search terms are inadequate for detecting implicit mentions, such as *Wouldn't it be better if I went now?* or *I would like to end the pain forever*.

- The number of possible (explicit) expressions is too large to capture effectively with keywords. Adding multiple or broader search terms inevitably increases the amount of false positives, adding to the burden for prevention workers who monitor the results. Even highly topical search terms yield false positives, e.g. *political suicide*.

- User-generated content tends to deviate from the linguistic norm. Typical problems include misspellings, the use of abbreviations, phonetic text and colloquial or ungrammatical language use. This may hinder keyword retrieval considerably (e.g. *siucide*).

In this paper, we present the first approach based on text classification to automatically detect suicide-related online content. The focus is on forum and blog messages in Dutch. Text classification of suicidal posts is a high-skew classification problem. To address the skew and data sparsity inherent to the problem, we investigate a wide range of potential features to model suicidality in text, and perform model optimisation through feature selection, hyperparameter tuning, and joint optimisation. The usability of the resulting system is evaluated on large datasets with realistic proportions of suicidal content.

## 2. Related research

Research conducted on the topic of 'suicidal text' has revolved primarily around suicide notes, arguably the most prototypical (albeit rare) textual expression of the suicide victim. For this reason, the genre has long been studied from psychological and psychiatric perspectives [38, 25, 37, e.g.]. The field recently saw the introduction of machine learning techniques: in [27], unsupervised clustering techniques are used to separate suicide notes from online newsgroup postings, and [31] applied supervised classification to distinguish genuine from fake notes. A corpus of 900 genuine suicide notes, annotated with fine-grained emotions, was released in the framework of the 2011 i2b2 NLP Challenge on emotion classification [30], allowing research on which emotions might be indicative of suicidal behavior, and how they can be found automatically.

Machine learning techniques have been applied in other areas of suicide research as well. [41] built a predictive model to identify patients at high risk from suicidal behaviour, using the information contained in electronic health records (EHR), such as administrative and demographic data, information on prior self-harm episodes and mental and physical health diagnoses. In addition to the clinical codes and numerical data, EHRs also contain free text (e.g. admission notes and discharge summaries), a source of unstructured information that is harder to take advantage of in data mining applications. [14] explored the use of NLP techniques to extract structured output from EHR notes, and used it in combination with clinical codes to detect potential relationships between drugs (e.g. antidepressants) or psychosocial stressors (e.g. depression, eating disorders, domestic abuse) to the incidence of suicidality. Models that incorporated information from free text were found to have much higher predictive value than those that only included clinical codes.

Work on the automatic detection of suicidal content in online media is scarce. [18] explored the possibility to identify bloggers at risk of suicide, by weighing profiles based on the occurrence of suicide-related keywords. The setup suffered from low precision (35% on the 20 highest-ranking profiles), and did not allow

to measure recall, i.e. the number of actually suicidal bloggers that are missing from the results.

A study by [19] also takes a keyword-based approach to detect at-risk content, on Twitter. Keywords were manually selected, along with exclusion terms (e.g. *cutting myself* and *shaving*, *accidentally* and *slack*). The approach was validated by collecting geolocated tweets that matched the terms, comparing them to tweets from random users from the same US state, and calculating the proportion of at-risk users versus background users. Proportions that departed from the expected (nation-wide) proportion were found to be strongly correlated to the actual age-adjusted state suicide rates, indicating that Twitter may be viable for large-scale monitoring of suicide risk factors. A limitation of the study is that it may not be reliable on an atomic level, i.e. for specific Twitter users.

In [29], suicide-related keywords were used to collect tweets with the Twitter search API. A sample of the resulting dataset was manually labeled as *strongly concerning*, *possibly concerning* or *safe to ignore*. Cross-validated machine learning models were found to perform as well as humans in distinguishing the categories, using token unigram bags-of-words as features. The study is the first to use machine learning to predict the level of concern for suicide-related messages. To find those suicide-related messages, keywords are still required.

The present study differs from the above work in that it does not rely on keyword filtering for the high-skew problem of detecting suicidal messages in general user-generated content. Instead, we investigate a supervised text classification approach with a rich set of text features. Performance is evaluated on an atomic level, so as to determine the practical feasibility to connect caregivers to potential victims. The experimental dataset allows to not only evaluate precision, but also recall and F-score.

## 3. Data

An important obstacle in using supervised machine learning instead of keyword spotting is that the former requires labeled training data, which in the case of suicidality detection is particularly hard to obtain. No corpora of suicide-related online content are readily available. We describe a newly developed scheme for suicidality annotation, and the collection of suicide-related and reference corpora for annotation, training and validation of online suicidality detection models.

### *3.1. Annotation*

Online text that mentions suicide or contains indications of suicidal thoughts can present itself in many forms, and not all of it is relevant for prevention purposes. In order to develop an annotation scheme that is motivated by practice, we collaborated with the Flemish Suicide Prevention Centre (CPZ[1]). This resulted in a cascaded scheme, based on criteria that are commonly used for suicide threat assessment [23]. Figure 1 presents an outline of the scheme.

First, a text is judged on its *relevance* using a clinical definition of suicide. It can either match the definition, mention suicide differently (in hyperboles or in non-clinical senses, e.g. suicide terrorism), or be unrelated. Only texts that match the definition are annotated further.

Next, the *genre* is annotated. Some texts are journalistic, informative or scientific (reports or research on suicide), others are personal in nature. For personal texts, we indicate whether they (partly) consist of a joke or other fictitious account, or one or more citations (e.g. the lyrics of a song).

In case of a non-fictitious personal text, the *subject* of the suicidal content is determined as either the author, some other person, or both. Instigations to commit suicide are flagged.

Finally, the *severity* of the suicide threat is annotated, depending on the presence of suicide thoughts or plans. Additionally, annotators can mark the

[1] http://www.preventiezelfdoding.be

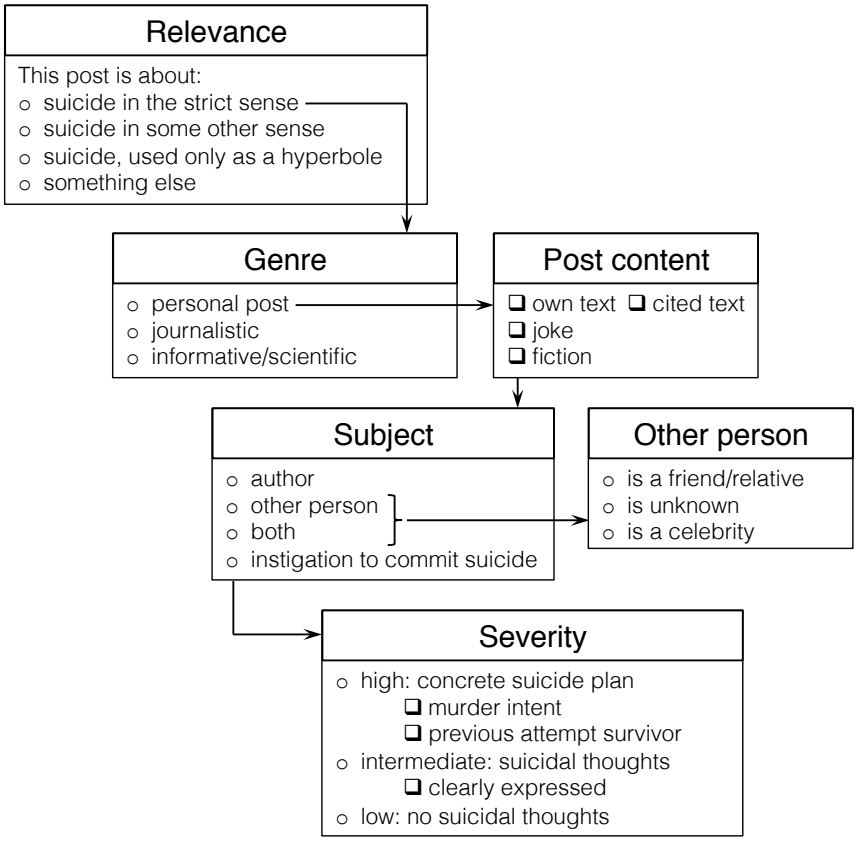


Figure 1: Schematic overview of text suicidality annotation. Round radio buttons indicate exclusive choices, square checkboxes indicate non-exclusive options.

*language* used to describe them, and indicate the presence of *risk factors* and *protective factors*.

The scheme was implemented in brat [39], an open-source online annotation tool which we modified to allow text-level annotations. A team of trained crisis responders at CPZ, consisting of four members of staff and two volunteers, carried out annotation of the experimental corpus described in Section 3.3, over the course of eight months. One member of staff managed the annotation effort and double-checked all annotations to remove errors, to ensure consistency and to resolve disagreements by discussing a consensus annotation. In cases of ambiguity, consensus erred on the side of caution and the more pessimistic

annotation was chosen.

### *3.2. Two detection tasks*

Based on the annotations, we defined two binary text classification problems that each correspond to a practical use case. The *relevance task* is concerned with the detection of suicide-related content, which includes all posts that receive a *Relevance: suicide in the strict sense* annotation. The *severity task* is about the detection of posts that present a high suicide risk, and should receive priority attention from suicide prevention workers. Positive instances for the severity task are the posts that have a *Severity: high* or *Severity: intermediate* annotation. This corresponds to the set of personally written, non-fictional posts that contain evidence (as per the annotator's judgment) that the post author or a known peer has suicidal thoughts and/or a suicide plan.

Inter-annotator agreement was assessed for both tasks, using a set of one hundred posts, forty of which contained suicide-related keywords. Two CPZ volunteers and one member of staff annotated the set independently. We calculated pairwise and average agreement, in terms of F-score (on the positive class) and Cohen's $\kappa$ [2]. IAA results are presented in Table 1.

| | | **A1-A2** | **A1-A3** | **A2-A3** | **Average** |
|---|---|---|---|---|---|
| **Relevance** | F-score | 0.9180 | 0.9062 | 0.8889 | 0.9044 |
| | $\kappa$ | 0.8821 | 0.8622 | 0.8380 | 0.8608 |
| **Severity** | F-score | 0.6923 | 0.6250 | 0.5455 | 0.6209 |
| | $\kappa$ | 0.6491 | 0.5946 | 0.5020 | 0.5819 |

Table 1: Pairwise and average inter-annotator agreement for the relevance and severity tasks. A1 and A2 are staff members, A3 is a volunteer annotator.

IAA F-scores are of interest for comparison to classifier performance, expressed with the same metric. They provide a rough estimate of the difficulty of the tasks for humans, and can therefore be viewed as a ceiling for performance of automatic classifiers, which infer their model from (imperfect) human

annotations. The $\kappa$ statistic is a widespread measure to evaluate agreement on labeling tasks. The average $\kappa$ of 0.8608 for the relevance task can be interpreted as good reliability ($\kappa > 0.8$). For the severity task, on the other hand, the average F-score of 62.09% and the moderate agreement (average $\kappa$ of 0.5819) suggest that this is a difficult task for humans, and automatic classification results also have to be interpreted in this light. The confusion can be explained by error percolation from preceding choices, and by differences in training and expertise between the annotators.

A qualitative analysis demonstrated that ambiguity is inherent to the annotation task, and to the medium: there are no infallible protocols for diagnosing suicide ideation, and the information that can be derived from a single social media message is limited. Annotation of severe suicide risk is especially difficult, as is reflected in the lower agreement scores. Confusion often stems from the ambiguous use of third person subjects (e.g. *some people can't cope anymore and they have to go! and then there is 1 exit: SUICIDE, that terrifying word!*). Some annotators consider these posts to be about some generic person, and therefore label them low-risk, while others interpret them as veiled expressions of suicide ideation by the author. Annotators may also need more information to judge whether suicidal thoughts are in play, because of vagueness by the author, or the limitations of a written and one-directional medium (compared to a spoken interaction).

Overall, we can conclude that given the inherent ambiguity of the task, the guidelines allow reliable annotation for relevance, and they are not the main cause of confusion for severity annotation.

### *3.3. Corpus collection*

The experiments in this study were performed on Dutch-language forum and blog messages posted on Netlog[2], a social networking site that was particularly popular amongst teenagers at the time of data collection. Given the low inci-

[2]http://nl.netlog.com/

dence of suicidal messages, we used a two-pronged approach to build a corpus containing a non-trivial amount of suicide-related text.

First, a sample of 1 040 posts containing a high percentage of suicide-related Netlog posts was obtained through the CPZ prevention centre. These posts had either been flagged as suicide-related, or matched a keyword search for *suicide* or its Dutch translations *zelfmoord* and *zelfdoding*. The average post contained 7.9 lines ($s = 11.0$), 121 tokens ($s = 78$) and 697 characters ($s = 419$). There is considerable deviation from the averages, with posts as short as 4 tokens. As is typical for social media content, overall post length is relatively short, although not as short as content from microblogging platforms such as Twitter.

After annotation, 82% ($n = 851$) of the posts were found to be about suicide in the strict sense, 2% about suicide in some other sense, 12% use the topic hypothetically and 5% are entirely unrelated to suicide. Following the definition for the relevance task, the annotated corpus therefore contains 851 relevant and 189 irrelevant posts. Since the majority of these irrelevant posts do contain references to suicide, distinguishing them from relevant posts is not a simple matter of keyword matching. For the severity task, posts with high ($n = 39$) and intermediate risk ($n = 218$) are pooled together, resulting in 257 severe posts.

The majority of the corpus ($n = 1\,000$) was expanded with 9 000 messages that were randomly sampled from a Netlog data dump from the same period. These messages were manually checked for presence of suicidality, and one additional relevant and severe post was found. This formed the training corpus ($n = 10\,000$) used for cross-validation experiments.

A small set of 40 relevant posts, 20 of which severe, was reserved for held-out and scaling experiments. It was combined with increasingly large samples from the Netlog data dump (10 000, 30 000, 100 000 and 300 000 posts), to obtain datasets that approach the real-world incidence of suicide-related material. The smallest resulting corpus ($n = 10,040$) serves as a held-out test set, and was manually checked for additional relevant ($n = 18$) and severe ($n = 6$) posts. The larger corpora were not manually annotated, so the labels for the majority

of their posts is unknown, although they can be assumed to predominantly unrelated to suicide. In the scaling experiments performed on these datasets, we therefore only measure precision. This provides insight into the practical usability (amount of noise) of our best models on highly skewed data.

Table 2 gives an overview of the size and label distribution for each of the corpora.

| **corpus** | **size** | **relevant** | | **severe** | | **not relevant** | **unknown** |
|---|---|---|---|---|---|---|---|
| train | 10,000 | 812 | 8.12% | 238 | 2.38% | 9,188 | - |
| test (held-out) | 10,040 | 58 | 0.58% | 26 | 0.26% | 9,982 | - |
| test (scaling) | 30,040 | 58 | 0.19% | 26 | 0.09% | 9,982 | 20,000 |
| test (scaling) | 100,040 | 58 | 0.06% | 26 | 0.03% | 9,982 | 90,000 |
| test (scaling) | 300,040 | 58 | 0.02% | 26 | 0.01% | 9,982 | 290,000 |

Table 2: Counts of different labels corresponding to each dataset

## 4. Text classification for suicidality detection

### *4.1. Feature representation*

Given the small amount of positive training material, preprocessing and feature design was oriented towards abstraction from the source text to decrease data sparsity. The raw input strings were converted with unidecode, a library to transliterate Unicode characters into ASCII and thus reduce variation. Next, the data was preprocessed with Pattern [5], to perform tokenisation (splitting off punctuation from words), part-of-speech tagging (assigning a morphosyntactic category to each token) and lemmatisation (leading to the base form of each token).

After preprocessing, we defined a set of features to model the two prediction tasks:

- *Bag-of-words features* consisting of word and lemma uni-, bi- and trigrams (W1, W2, W3, and LEM1, LEM2, LEM3). We also included character bi-,

tri- and fourgrams for both the words (WCH2, WCH3, WCH4) and lemmas (LCH2, LCH3, LCH4) as we expect them to be more robust to noise, like orthographic variation, than token-based representations.

- *Polarity lexicon features.* Because we suspect that negative polarity in a post might be correlated with suicidality, we implemented the following polarity features based on two subjectivity lexicons available for Dutch [21, 5], and one lexicon for emoticons [24]: the ratio of matched positive or negative tokens in a document (PAT-ratio+, PAT-ratio-, DUO-ratio+, DUO-ratio-, EMO-ratio+, EMO-ratio-), the sum of polarity scores of all matched lexicon entries (PAT-sum, DUO-sum, EMO-sum), and the raw positive and negative counts for emoticons (EMO-count+, EMO-count-). The PAT and DUO features were also calculated on the last 10 tokens of a post, since those might provide a summary of its emotional orientation.

- *Domain-specific lexicon features* were extracted through automatic terminology extraction [26] from a corpus of 290 transcripts from the CPZ emergency chat hotline. We included three types of term features: an exact match feature (TERM-exact) and two more relaxed variants for multiword terms, allowing for random word ordering either in a context of 5 words (TERM-local), or in the entire post (TERM-global).

- *Topic model features* for discovering semantically related words which are not captured by the BoW features. In absence of a large background corpus containing suicidal material, we used BootCaT [1] to crawl a corpus of web documents about suicide. As seed terms, we used the most frequent terms which were extracted from the chat transcripts, leading to 105 search terms that were used by BootCaT to retrieve 50 pages for each query. The resulting background corpus, containing over two million words, and the previously mentioned chat transcripts corpus were then fed to Gensim [33] for the construction of latent semantic topic models. We derived two types of LSA features: the $k$ individual topic scores of the document ($k$ = 20, 50, 100, 200) (LSA-20, LSA-50, LSA-100, LSA-200) and the average similarity

between a document and the 290 documents in the chat transcripts corpus (LSA-20-avg, LSA-50-avg, LSA-100-avg, LSA-200-avg).

- *Surface features* describing the basic surface properties of the original text such as post length (LENGTH), ratio of capitalised characters (CAPS-char), ratio of tokens with more than one capitalised letter (CAPS-token).

- *Named entity features*, extracted with DBPedia Spotlight [28]. We hypothesized that the presence of names of people, organisations, etc. could help in recognising journalistic and informative texts, as well as personal texts about celebrities. Therefore, we added three features based on the DBPedia ontology linking: one binary feature indicating the presence of one or more NEs in a post (NE-presence), and two integer features indicating the number of (unique) named entities (NE-count, NE-unique).

The resulting feature vectors consist of 1,934,186 individual features, the bulk of which ($> 99.9\%$) are binary BoW features. In Section 5.1, we discuss how feature vector size was reduced.

### *4.2. Learning algorithm*

In the text classification literature, support vector machines (SVMs) and Naive Bayes (NB) are commonly used. When properly tuned, they have been observed to achieve similar to better performance compared to more complex algorithms, and are typically sufficient for solving practical text categorisation problems [22, 36]. In a controlled study on common text categorisation methods by [46], SVMs were found to be robust in dealing with skewed category distributions.

In our experiments, we use LIBSVM[3], version 3.17 [3]. Since the hyperparameters of a learning algorithm can have a dramatic impact on performance, a variety of SVM settings was experimentally explored:

[3] `http://www.csie.ntu.edu.tw/~cjlin/libsvm`

- Linear, polynomial and sigmoid kernels. We omit the RBF kernel, which is less suitable for the high-dimensional feature vectors typical of text classification [17].
- Soft margins cost parameter $C$ ($2^{-6}$ to $2^{12}$, stepping by a factor of 4)
- For non-linear kernels, we varied the free parameter $\gamma$ between $2^{-14}$ and $2^4$ (stepping by a factor of 4), and the polynomial degree $d$ between 2 and 5. We expect better results for lower degrees of freedom, since larger degrees tend to overfit on NLP problems [13].

All data sets were scaled before applying SVM, i.e. all feature values were linearly mapped to the range $[0, 1]$, using the svm-scale utility bundled with LIBSVM.

### *4.3. Evaluation*

We evaluated models in terms of F-score on the positive class. Given the skewness of the detection task, we consider other metrics less suitable. Whereas F-score is affected by skew in only one direction (for the majority class), Cohen's kappa is affected in both directions [20]. This makes it less interpretable for comparing performance on datasets that have different levels of skew (e.g. between the cross-validation and the held-out datasets). A rank metric such as area under the receiver operating characteristic curve (AUC ROC) is unaffected by skew, but for a minority class detection task with strong skew, it would fail to intuitively show the burden of false-positive predictions, because it compares them to the total amount of instances (false positive rate), rather than to the amount of true positives (precision). For this reason, [34] argues against the use of AUC ROC with strongly imbalanced datasets in which the number of negatives outweighs the number of positives significantly.

We report F-scores with $\beta = 1$, resulting in a harmonic mean of precision and recall, but also with $\beta = 2$. The latter gives more weight to recall, which is important in this domain: false negatives (not detecting a potentially suicidal post) are more problematic than false positives. In an application of our task,

where posts are automatically filtered for review by a suicide prevention worker, false positives can still be ignored by the user, whereas false negatives would not be presented. A risk of overemphasizing recall is that moderation could become ineffective when there are too many false alarms.

All models are evaluated using tenfold cross-validation on the training corpus of 10 000 posts. Four models with optimal features and hyperparameters are trained on all the training data, and used for held-out testing, scaling experiments and error analysis.

As a baseline, we report scores of an SVM classifier with default hyperparameters (linear kernel, $C = 1$), which exclusively relies on token unigrams.

## 5. Model optimisation

Optimisation, as argued in [16], is an essential exploration of the space of possible experiments, and allows reliable conclusions to be drawn about the performance of a given machine learning method exploiting a given set of features.

### *5.1. Feature selection*

Since the tasks of detecting suicide-related and severe messages in user-generated content are novel, we do not know from previous work which type of information is useful for accurate classification. However, it is unlikely that the full feature vectors of almost two million individual features will produce the best results. Therefore, we experimented with two types of feature selection: a *filter approach* in which feature selection is done independently of classifier performance and a *wrapper approach* where classifier performance guides the selection.

*Feature filtering.* With a filter approach to feature selection, an evaluation function is used to score each feature's informativeness for a given task, without explicitly testing the features with a learning algorithm. Selection can be done by keeping the $n$ features with the highest score, or by removing features that score below a given threshold. There are a number of metrics available to perform

this selection, including *information gain*, *gain ratio*, *chi-squared*, *document frequency*, *mutual information*, *odds ratio* and *binormal separation*. Based on the benchmark studies of [47] and [8], we opted to used information gain. The threshold to filter features was set heuristically to 0.001, so that the number of features for both tasks would be around 20 000, a dimensionality reduction of two orders of magnitude. This resulted in 21 791 features for the relevance task and 9,351 features for the severity task.

*Wrapped feature selection.* Wrapper methods determine the informativeness of a feature set by validating it with the intended learning algorithm. The main advantages of this approach are that it selects the optimal features for a specific problem and learner, rather than using a heuristic metric to estimate feature salience, and that it test combinations of features rather than features in isolation, so that feature interactions and redundancies are considered. As opposed to the aforementioned filtering methods, where individually scoring each feature takes linear time ($O(n)$ where $n$ is the number of features), a wrapper method would take exponential time ($O(2^n)$) if an exhaustive feature subset search were performed. This disadvantage is compounded by the fact that evaluating a single combination (whereby a model needs to be trained and tested) is computationally much more involved than calculating a filtering metric. We defined three ways of partitioning the features:

- *group* selection, where each of the 46 groups is either entirely included or excluded
- *nbest* feature group selection, in which the number of features in each group is limited to the 500 best features, according to the information gain metric
- *stratified* selection, whereby each feature group is sorted by IG and split into a number of strata. The number of strata is varied taking into account the differences in feature group size. We define the number of strata $S_i$ for feature group $i$ as a function of its feature count $n_i$, by rounding the

cube root of group size to the nearest integer:

$$S_i = round(\sqrt[3]{n_i}) \tag{1}$$

The motivation for using the cube root is that it provides a good tradeoff between granularity and number of strata. Small feature groups ($1 \leq n_i < 100$) will be split into a small number of fine-grained strata ($1 \leq S_i \leq 5$). As feature group size goes up, they are split into more bins, but the number of bins grows slowly. This prevents the search space from becoming too large, but comes at the expense of granularity. A group with 1 000 features, for example, will be split into 10 bins of size 100. Using this binning strategy, we obtain 187 stratified groups for the relevance task, and 154 for severity.

*5.2. Hyperparameter optimisation*

The hyperparameters of the learner can have an influence on practical aspects of running the algorithm, such as speed or required memory, but can also affect performance. We therefore performed hyperparameter optimisation so as to minimise the training error. Hyperparameters like the cost value $C$ for SVM, for example, influence the capacity of a learner to fit the training data, and can be tuned with the goal of preventing underfitting (the model does not capture underlying trends in the training data) and overfitting (the model is overly complex and fits noise in the data), so as to achieve good generalisation. For the task of detecting high-risk suicidal content, for example, a model suffering from overfitting would only be capable of detecting posts with features (e.g. words) that are very similar to the ones found in specific positive training instances, and suffer from low recall as a result.

Since we do not know beforehand which hyperparameter combination is optimal for the two classification tasks, we varied the following hyperparameters: we allow 3 kernels, 10 cost values $C$, 10 $\gamma$ values and 4 degrees of freedom $d$. Considering the compatibility of the kernels with the other hyperparameters, the following amounts of combinations are possible: 10 ($C$) for linear kernels,

$10 \times 10 \times 4 = 400$ ($C \times \gamma \times d$) for polynomial kernels and $10 \times 10 = 100$ ($C \times \gamma$) for sigmoid kernels, making a total of 510 possible combinations.

In experiments where no hyperparameter optimisation is applied, we use the LIBSVM default settings: a linear kernel with $C = 1$.

### *5.3. Parallel optimisation with genetic algorithms*

Hyperparameter optimisation and feature selection each present a search problem that needs to be solved. Since both optimisation steps can also interact, we performed joint optimisation in which both problems are considered at the same time. Two possible approaches to tackling this search problem are manual tuning in which different combinations are manually evaluated, or grid search, which is an exhaustive search method. In the case of feature selection, hillclimbing has also long been a popular search procedure, but it is sensitive to local optima.

We opted to use genetic algorithms [15, 11, 44] for the joint optimisation. They have been shown to work well for jointly estimating features and hyperparameters for SVM [4] and offer the advantage that optimisation is initialised from a variety of points in the search space. Evolutionary algorithms borrow the concepts of fitness-based selection, mutation, inheritance and evolution, and apply them to a search problem. First, the search space is represented as a genome of fixed length. In the case of joint feature selection and hyperparameter optimisation, the genome will consist of one binary-valued gene for each feature group (with value 1 if the feature group is selected, 0 if it is not), and one multi-valued gene per hyperparameter (see Figure 2 for an example). The exact size of the search space for each optimisation run is described in the third column of Tables 3 to 7.

An initial population is created containing a fixed number of individuals. Next, the fitness of all individuals in the population is evaluated using a fitness function, in our case F-score. If the termination criterion (e.g. stop when the highest fitness has not changed in five generations) has not been satisfied, a new population of individuals is created relying on mechanisms such as selec-

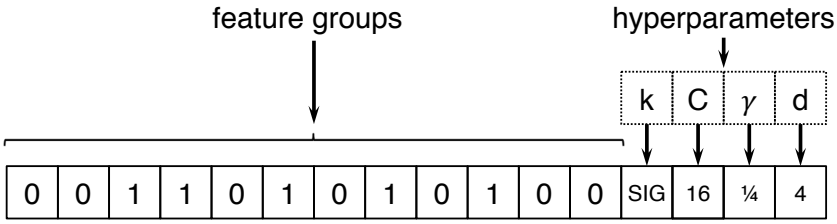


Figure 2: A potential solution to the problem of joint optimisation in LIBSVM, with 5 selected feature groups and a sigmoid kernel. The structure of this individual is dictated by the genome, the genetic representation of the search space, consisting here of 12 bits for the selection of feature groups, and 4 hyperparameters.

tion, mutation and crossover. The evolution continues until termination. With genetic algorithms, we have at our disposal a means of finding solutions in a large search space. It is much more efficient than e.g. testing every possible solution, but the computation time $t$ required to evaluate a single candidate solution is still quite significant. On a single 3.5 GHz core, for example, doing tenfold cross-validation on our experimental dataset (with all features and default hyperparameters) takes in the order of hours for LIBSVM.

It should therefore not surprise that fitness calculation is the most time-consuming step in a GA search. In order to reduce the overall computation time, the **genetic algorithm toolbox Gallop** [7] was developed to run the optimisation in parallel. It is a Python library based on DEAP[4], the Distributed Evolutionary Algorithms in Python framework [9]. Gallop provides the functionality to wrap a complex optimisation problem as a genome, and to distribute the computational load of the GA run over multiple processors or to a high performance computing cluster. When a population is created or offspring produced, Gallop builds genotypes with the available hyperparameter options, and checks them for compatibility. Incompatible options are disabled. With a linear SVM kernel, for example, the $\gamma$ and $d$ hyperparameters are removed. Gallop

[4]http://deap.gel.ulaval.ca/

supports individual and grouped feature selection, and selected features or feature groups are represented as bits in the genome. The top-level GA process is implemented in the DEAP framework. It keeps track of the current population and its history (so that identical individuals are only evaluated once), and handles selection and reproduction. The population history is stored after each generation. This allows for *checkpointing*, resuming the GA run after an error or restarting it with different termination settings.

For our experiments, Gallop was run on a Tier-2 supercomputer. Each generation was submitted as an array of job requests to be processed simultaneously, and Gallop polls the cluster until all jobs are finished. The population size of 100 was set at the low end of what is generally recommended, given the computationally expensive validation procedure: the fitness of each individual was determined using tenfold cross-validation on the full training set. We used single-point crossover with a probability of 0.9, and a mutation rate of 0.3. These settings are both relatively high to promote exploration, which can compensate for the small population size and avoid premature convergence. We applied elite selection at a rate of 0.1, i.e. promoting the fittest 10% of a population directly to the next generation. For the remaining 90%, we used tournament selection with a tournament size of three. Roulette wheel selection is significantly slower than other methods, and truncation selection offers little exploration [12]. Tournament selection provides a good trade-off between speed and exploration when the tournament size is sufficiently small.

Evolution was terminated after 50 generations, or when the best fitness had changed less than 0.0001 over the last 5 generations. In practice, all optimisation runs converged before reaching the maximum number of generations. As an example, Figure 3 shows the convergence of the 7 optimisation runs towards $F_1$ for the relevance task, with runs taking between 6 and 27 generations before satisfying the termination criterion. In terms of computational effort, the evaluation of one generation required an average wall time of around 3200 seconds on the computing cluster, with the individuals being evaluated in parallel over 100 2.6GHz cores with 6GB RAM each. A 26-generation optimisation run therefore

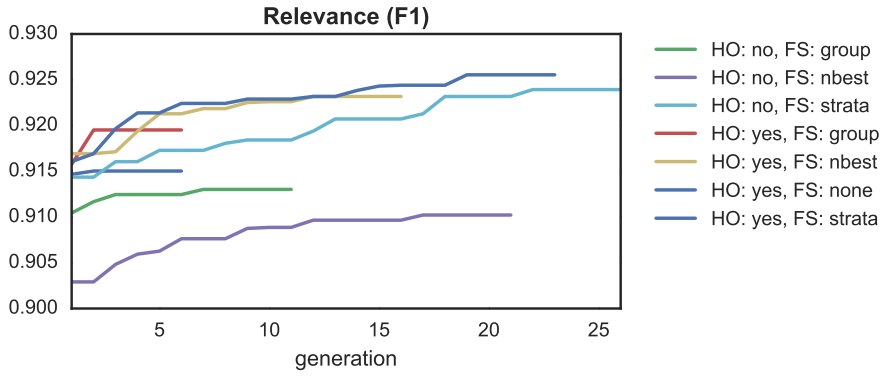


Figure 3: Evolution of the maximum fitness score ($F_1$) per generation, over the course of the optimisation runs for the relevance task (HO = hyperparameter optimisation, FS = feature selection).

required about 2300 compute hours.

## 6. Results and discussion

In this chapter, we start by presenting the results of the optimisation experiments with cross-validation on the training set ($n = 10,000$). We describe performance on the relevance and severity detection tasks (6.1 and 6.2), and how it differs between optimisation strategies. The effects of optimisation are discussed in more detail in Section 6.3, and selected feature groups in Section 6.4.

Using the four best classifiers from the cross-validation experiments, we discuss learning curve behaviour and how performance is affected on the highly skewed held-out test set ($n = 10,000$) in Section 6.5). Finally, the scaling datasets (up to $n = 300,000$) are used to further increase skew and approach the real-world incidence of alarming posts. We do a qualitative analysis of the positive predictions from the systems and discuss usability in Section 6.6.

### *6.1. Relevance task*

For the detection of suicide-related posts, the classification objective is to label a post as either relevant or not. Tables 3 and 4 list the results of two sets

| HO | FS | GA search space | $F_1$ | $F_2$ | Prec. | Rec. |
|---|---|---|---|---|---|---|
| baseline | | $n/a$ | 87.65 | 86.18 | 90.22 | 85.22 |
| no | none | $n/a$ | 90.61 | 90.48 | 90.84 | 90.39 |
| | group | $2^{46}$ | 91.30 | 91.20 | 91.47 | 91.13 |
| | nbest | $2^{46}$ | 91.02 | 90.72 | 91.53 | 90.52 |
| | strata | $2^{187}$ | 92.39 | **92.59** | 92.05 | 92.73 |
| yes | none | 510 | 91.50 | 91.50 | 91.50 | 91.50 |
| | group | $510 \times 2^{46}$ | 91.95 | 92.49 | 91.06 | **92.86** |
| | nbest | $510 \times 2^{46}$ | 92.32 | 92.42 | 92.15 | 92.49 |
| | strata | $510 \times 2^{187}$ | **92.55** | **92.59** | **92.50** | 92.61 |

Table 3: Relevance classification scores on the training set, optimised towards $F_1$ (HO = hyperparameter optimisation, FS = feature selection, GA = genetic algorithm).

| HO | FS | GA search space | $F_1$ | $F_2$ | Prec. | Rec. |
|---|---|---|---|---|---|---|
| baseline | | $n/a$ | 87.65 | 86.18 | 90.22 | 85.22 |
| no | none | $n/a$ | 90.61 | 90.48 | 90.84 | 90.39 |
| | group | $2^{46}$ | 91.31 | 91.28 | 91.37 | 91.26 |
| | nbest | $2^{46}$ | 91.04 | 90.87 | 91.33 | 90.76 |
| | strata | $2^{187}$ | 92.59 | 92.89 | **92.08** | 93.10 |
| yes | none | 510 | 91.47 | 92.81 | 89.32 | 93.72 |
| | group | $510 \times 2^{46}$ | 92.31 | 93.22 | 90.82 | 93.84 |
| | nbest | $510 \times 2^{46}$ | 92.27 | **93.35** | 90.52 | **94.09** |
| | strata | $510 \times 2^{187}$ | **92.69** | 93.31 | 91.69 | 93.72 |

Table 4: Relevance classification scores on the training set, optimised towards $F_2$ (HO = hyperparameter optimisation, FS = feature selection, GA = genetic algorithm).

| | baseline | FS: none, HO: no | FS: none, HO: yes | FS: group, HO: no | FS: group, HO: yes | FS: nbest, HO: no | FS: nbest, HO: yes | FS: strata, HO: no | FS: strata, HO: yes |
|---|---|---|---|---|---|---|---|---|---|
| baseline | - | *** | *** | *** | *** | *** | *** | *** | *** |
| FS: none, HO: no | *** | - | | * | * | | *** | *** | *** |
| FS: none, HO: yes | *** | | - | | | | | | *** |
| FS: group, HO: no | *** | * | | - | | | | * | * |
| FS: group, HO: yes | *** | * | * | | - | | | | |
| FS: nbest, HO: no | *** | | | | | - | * | * | * |
| FS: nbest, HO: yes | *** | * | * | | | | - | | |
| FS: strata, HO: no | *** | *** | * | *** | | * | | - | |
| FS: strata, HO: yes | *** | *** | *** | * | | * | | | - |

Table 5: Significance of pairwise difference between system outputs for relevance, $*$ <= 0.05, $***$ <= 0.0014 (Bonferroni-adjusted). Above diagonal: F1-optimised systems, below diagonal: F2-optimised systems.

of cross-validation experiments. The results of the baseline system (SVM with token unigrams only) are displayed in the first row. The second row shows the results obtained with a LIBSVM classifier configured to use the default hyperparameters and all features, i.e. the unoptimised results. The next seven rows each represent a separate Gallop optimisation run, with various optimisation settings: with or without hyperparameter optimisation (HO) and with none or one of the three feature selection (FS) strategies. For these optimised runs, we display the scores of an elite individual, i.e. a classifier with settings optimised towards a particular fitness score. The classifiers in Table 3 were optimised towards $F_1$, those in Table 4 towards $F_2$.

To determine whether the difference between a pair of systems is statistically significant, we applied two-tailed binomial testing on their outputs [35]. Table 5 shows the results of these pairwise comparisons, for a regular significance level of

0.05 ($*$) and a Bonferroni-adjusted significance level of $0.05/36 \approx 0.0014$ ($***$), since 9 systems entail 36 pairwise comparisons. Comparisons between the F1-optimised systems are shown above the diagonal, those between F2-optimised systems below the diagonal.

A first observation is that text classification is a viable and promising strategy for detecting social media posts that are about suicide. The tenfold cross-validation experiments on the training data show that all classifiers significantly outperform the unigram baseline, which mainly suffers from lower recall scores compared to the other models.

The systems obtained after optimisation search with genetic algorithms results in better scores, regardless of the fitness objective. However, not all of these differences are strongly significant. Compared to the unoptimised system with all features, stratified feature selection always results in significantly better systems. When it is combined with hyperparameter optimisation, it also significantly outperforms the system with HO and all features.

The best-performing model, obtained after joint optimisation with stratified feature groups, achieves an $F_1$ score of 92.69%, and offers a good balance between precision and recall. The best $F_1$ value is obtained in an optimisation towards $F_2$, although the best score from an $F_1$-optimised system (92.55%) is not statistically different. Both systems use the same stratified setup with hyperparameter optimisation.

### *6.2. Severity task*

Posts that contain a severe threat of suicide are complex to detect, both for human annotators and machine learning models. The scores shown in Tables 6 and 7 are considerably lower than those for the relevance task. This is not surprising, given that humans are also puzzled more by the ambiguity inherent to this task (reflected in lower inter-annotator agreement scores in Table 1), and the smaller amount of training material.

Whereas for the relevance task all systems (optimised or not), were found to significantly outperform the baseline, for severity this is only true when there is

| HO | FS | GA search space | $F_1$ | $F_2$ | Prec. | Rec. |
|---|---|---|---|---|---|---|
| baseline | | $n/a$ | 55.40 | 51.75 | 62.77 | 49.58 |
| no | none | $n/a$ | 61.36 | 57.40 | 69.31 | 55.04 |
| | group | $2^{44}$ | **69.04** | 63.93 | 79.67 | 60.92 |
| | nbest | $2^{44}$ | 67.13 | 62.99 | 75.39 | 60.50 |
| | strata | $2^{154}$ | 67.29 | 62.77 | 76.47 | 60.08 |
| yes | none | 510 | 61.36 | 57.40 | 69.31 | 55.04 |
| | group | $510 \times 2^{44}$ | 68.88 | 63.87 | 79.23 | 60.92 |
| | nbest | $510 \times 2^{44}$ | 68.54 | **64.03** | 77.66 | **61.34** |
| | strata | $510 \times 2^{154}$ | 67.33 | 60.25 | **83.75** | 56.30 |

Table 6: Severity classification scores on the training set, optimised towards $F_1$ (HO = hyperparameter optimisation, FS = feature selection, GA = genetic algorithm).

| HO | FS | GA search space | $F_1$ | $F_2$ | Prec. | Rec. |
|---|---|---|---|---|---|---|
| baseline | | $n/a$ | 55.40 | 51.75 | 62.77 | 49.58 |
| no | none | $n/a$ | 61.36 | 57.40 | 69.31 | 55.04 |
| | group | $2^{44}$ | 66.82 | 62.88 | 74.61 | 60.50 |
| | nbest | $2^{44}$ | 65.95 | 64.94 | 67.70 | 64.29 |
| | strata | $2^{154}$ | **69.51** | **66.81** | 74.52 | **65.13** |
| yes | none | 510 | 61.36 | 57.40 | 69.31 | 55.04 |
| | group | $510 \times 2^{44}$ | 67.29 | 62.50 | **77.17** | 59.66 |
| | nbest | $510 \times 2^{44}$ | 68.92 | 66.07 | 74.27 | 64.29 |
| | strata | $510 \times 2^{154}$ | 66.96 | 64.81 | 70.89 | 63.45 |

Table 7: Severity classification scores on the training set, optimised towards $F_2$ (HO = hyperparameter optimisation, FS = feature selection, GA = genetic algorithm).

| | baseline | FS: none, HO: no | FS: none, HO: yes | FS: group, HO: no | FS: group, HO: yes | FS: nbest, HO: no | FS: nbest, HO: yes | FS: strata, HO: no | FS: strata, HO: yes |
|---|---|---|---|---|---|---|---|---|---|
| baseline | - | * | * | *** | *** | *** | *** | *** | *** |
| FS: none, HO: no | * | - | | *** | *** | * | *** | *** | *** |
| FS: none, HO: yes | * | | - | *** | *** | * | *** | *** | *** |
| FS: group, HO: no | *** | * | * | - | | | | | |
| FS: group, HO: yes | *** | * | * | | - | | | | |
| FS: nbest, HO: no | * | | | | | - | | | |
| FS: nbest, HO: yes | *** | * | * | | | | - | | |
| FS: strata, HO: no | *** | *** | *** | | | * | | - | |
| FS: strata, HO: yes | *** | | | | | | | | - |

Table 8: Significance of pairwise difference between system outputs for severity, $*$ <= 0.05, $***$ <= 0.0014 (Bonferroni-adjusted). Above diagonal: F1-optimised systems, below diagonal: F2-optimised systems.

feature selection (see Table 8). When optimising towards $F_2$, nbest feature selection needs to be combined with hyperparameter optimisation to significantly beat the baseline. Between the optimised systems, adding feature selection always brings improvement over the systems with all features (regardless of hyperparameter optimisation), and in most cases this improvement is significant.

The best $F_1$ score of 69.51% is obtained with stratified feature group selection. The system finds 2 out of 3 severe posts, and only 1 in 4 suggested posts is not severe. From a usability perspective, this is very reasonable in terms of noise, and can be considered a step forward in automated prevention practice. Nevertheless, better recall is desirable.

*6.3. Effects of optimisation*

The results indicate that the genetic algorithm approach to optimise the selected features and hyperparameters is effective: optimisation invariably improves performance, with error reductions of up to 25% for both tasks. Most of these improvements are strongly significant compared to the baseline, and the choice of feature selection method and inclusion of hyperparameter optimisation can have a significant impact.

We optimised towards two fitness objectives: $F_1$, and $F_2$ for improved recall. For both tasks, optimisation towards $F_2$ often yields the best overall $F_1$ and $F_2$ score. We hypothesize that optimising towards recall is the better strategy for this task. All classifiers optimised for $F_1$ obtain a score that is balanced in terms of precision and recall, whereas $F_2$ classifiers consistently achieve lower precision and higher recall. In other words, the different optimisation objectives reliably steer the GA in the preferred direction, but the aim for better recall eventually leads to the best $F_1$ scores as well. It is plausible that $F_1$ optimisation discards sub-optimal solutions with high recall before they can be fine-tuned for better precision. Rather than optimising towards a single objective function, it would be beneficial to optimise precision and recall simultaneously and find solutions spread along the full Pareto-optimal front. Experiments with multiobjective genetic algorithms like NSGA-II [6] are a promising avenue for future work.

Hyperparameter and joint optimisation can make the difference between a pair of classifiers statistically significant. Nbest feature group selection, for example, significantly outperforms no feature selection only when it is combined with hyperparameter optimisation (for both tasks). For the relevance task, tuning the hyperparameters leads to better performance, especially in terms of recall. For severity, hyperparameters have a less predictable impact, and including them for optimisation can even deteriorate the optimal results. We found this to be caused by search space sparsity, which can be remedied by increasing the population size for the genetic algorithm.

Unlike hyperparameter optimisation, feature selection is always effective. Of the three tested strategies for feature selection, stratified feature group selection

performed best. It offers more granularity by splitting large feature groups into ranked bins. The selection results demonstrate that this is beneficial: in the ngram feature groups, for example, more than half of the bins is removed. Not only does this result in better scores, it also makes for a model that requires fewer features. Furthermore, we find that strata are selected from all stratified feature groups. Instead of having to include or exclude entire groups, as is the case with *group* and *nbest* selection, the search algorithm can pick the most useful subsets of a feature group.

### *6.4. Selected features*

We defined a variety of features with the aim of gaining an insight into what kind of information is relevant for suicidality modelling. Overall, we find that virtually all feature groups are informative to some extent. More specifically, Table 9 (relevance) and 10 (severity) show how often each feature group was selected in the top individuals at the end of an optimisation run. Selection status is shown for the *group* and *nbest* selection methods. Tables for stratified selection are omitted for brevity (since they contain many more groups), but the same trends persist. The following observations can be made:

- Both token and character bag-of-words features are often selected. We notice that ngrams based on the original words are mutually interchangeable with those based on lemmas. For the relevance task, token unigrams and bigrams are preferred, whereas for severity, there is a clear preference for longer ngrams: trigrams are selected, unigrams are discarded. This would indicate that relevant posts can be successfully identified with short keywords, whereas the added specificity of collocations is required for severity detection.

- Term features with non-exact matching are always included. This validates the approach of extracting highly salient collocations from a specialised corpus. Relaxed term matching also provides better abstraction than the token ngram or exact term matching features.

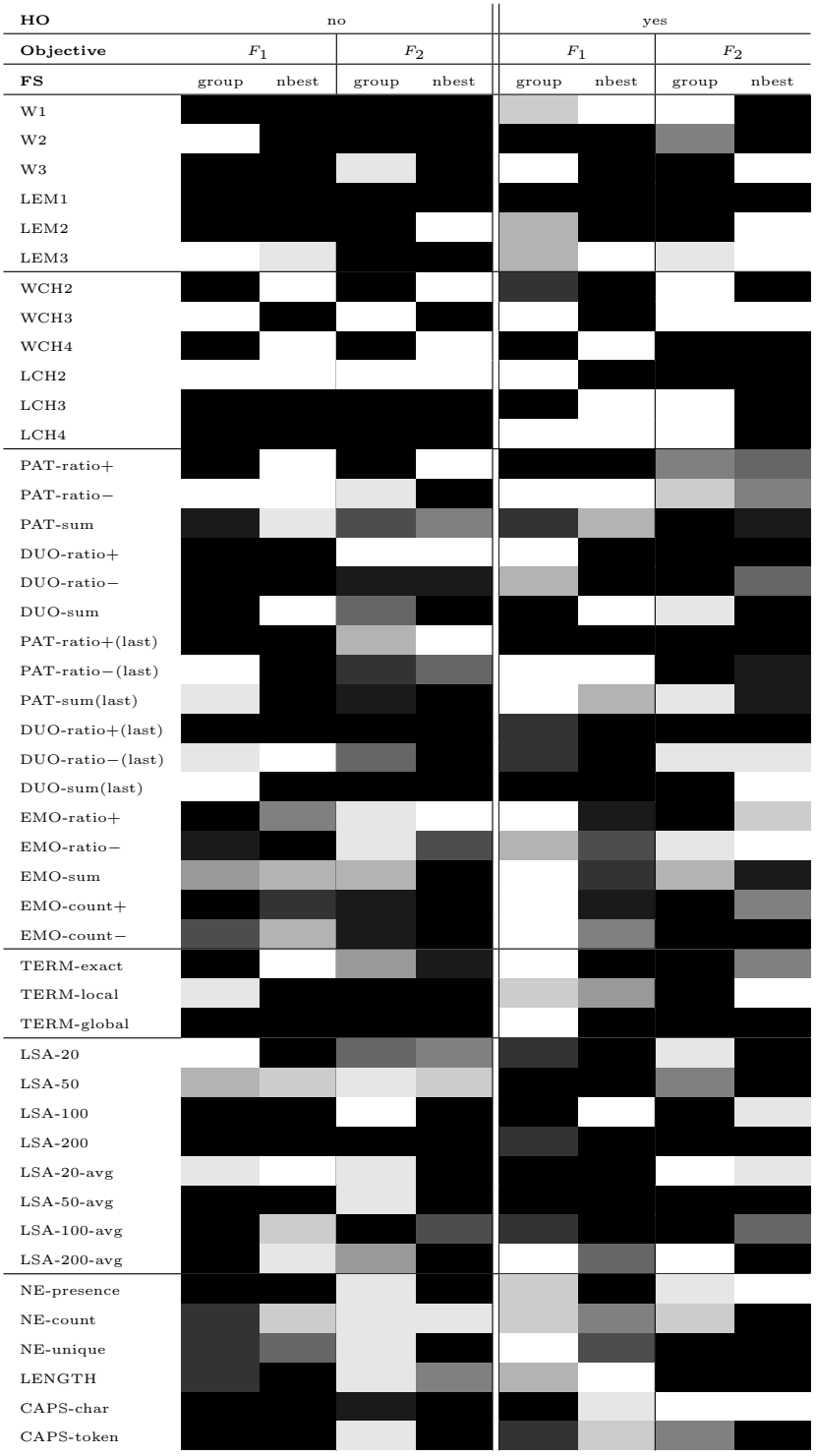


Table 9: Feature group selection status in all relevance models with regular or nbest feature group selection (FS), with or without hyperparameter optimisation (HO), and optimised towards $F_1$ or $F_2$. Cell colour indicates the relative frequency of selection (darker = more often selected).

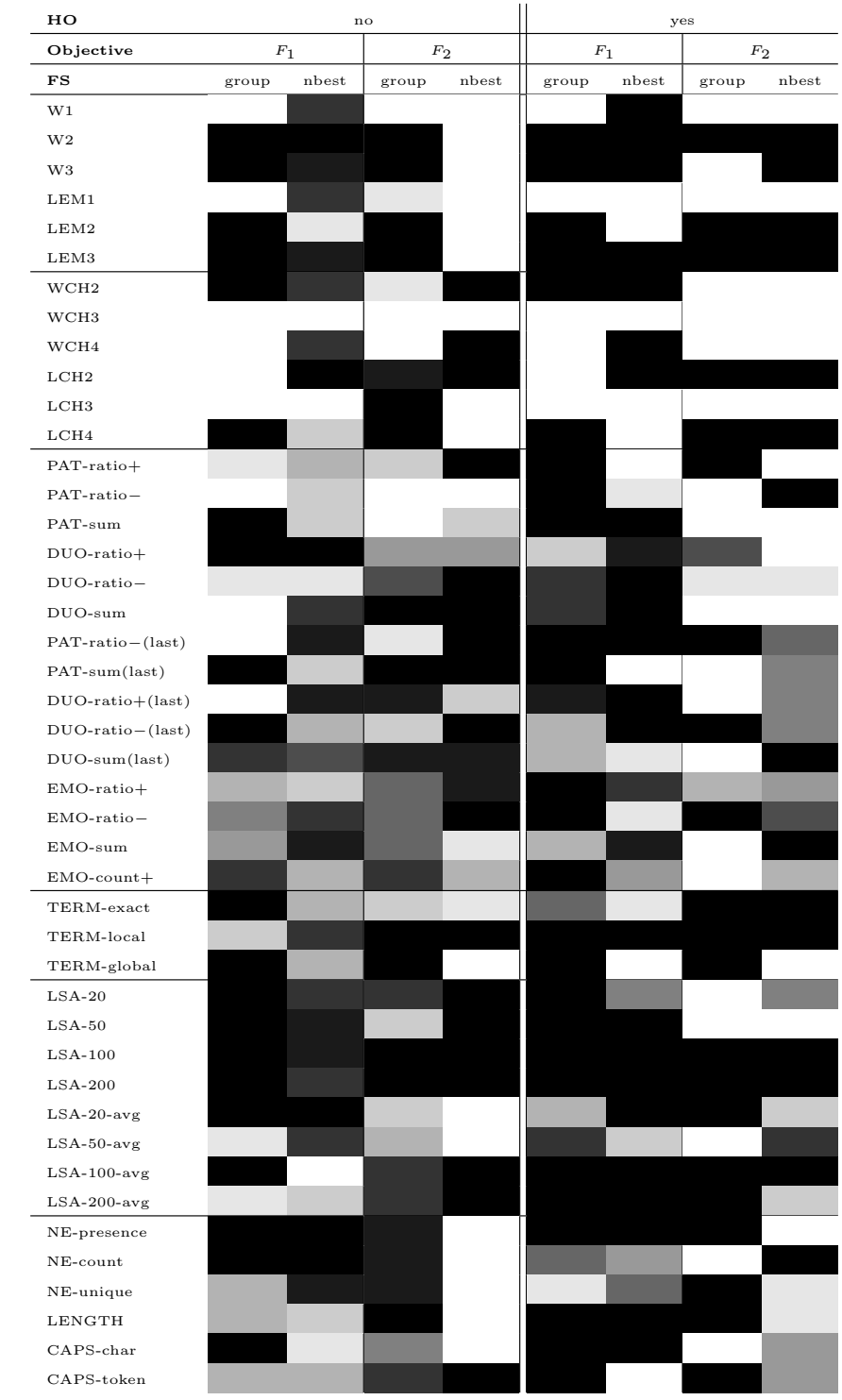


Table 10: Feature group selection status in all severity models with regular or nbest feature group selection (FS), with or without hyperparameter optimisation (HO), and optimised towards $F_1$ or $F_2$. Cell colour indicates the relative frequency of selection (darker = more often selected).

- The abstraction obtained by clustering semantically related concepts into topics is beneficial. LSA features are found to perform very well, particularly for severity. Features with high amounts of topics are favoured, indicating that high topic granularity is most adequate to detect signals of suicidality.

- The assumption that negative (or lack of positive) polarity is associated with posts about suicide is confirmed. Features from the polarity lexicons are selected for both tasks. Additionally, we find that the polarity of the final words in a message is most informative.

- The miscellaneous feature groups are selected least often. For the severity task, named entity information is salient. We speculate that these features may help in labeling informative and journalistic messages as non-severe.

### *6.5. Held-out testing and learning curves*

We selected the best classifiers per task and per optimisation objective (four in total) for training on the entire dataset using their optimal hyperparameters and features. In this section, we discuss how they behave on a held-out test set that has much higher skew than the training set. Both sets contain $10,000$ instances, but support in the held-out set is just 58 for the relevance task (training set: 812), and 26 for severity (training set: 238). Given the limited support, quantitative results allow some observations, but they should be interpreted with caution. We perform a qualitative analysis of the results on this held-out test in Section 6.6.

The held-out results for relevance (Table 11) show $F_1$-scores of around 75%, with high precision at over 97%. Recall drops considerably compared to the cross-validation experiments, from above 90% to around 60%. This can be partly explained by the higher proportion of severe posts in the relevant sample (almost half), since severe posts generally contain more implicit references to distress. For the severity task, results from the two selected systems differ considerably: the $F_1$-optimised system obtains the better $F_1$-score of 56.41%,

| Task | System | $F_1$ | $F_2$ | Prec. | Rec. |
|---|---|---|---|---|---|
| Relevance | FS: strata, HO: yes, obj: F1 | 75.79 | 66.92 | 97.30 | 62.07 |
| Relevance | FS: nbest, HO: yes, obj: F2 | 74.47 | 65.30 | 97.22 | 60.35 |
| Severity | FS: group, HO: no, obj: F1 | 56.41 | 47.01 | 84.62 | 42.31 |
| Severity | FS: strata, HO: no, obj: F2 | 37.21 | 33.06 | 47.06 | 30.77 |

Table 11: Classification scores on the held-out test set for four selected classifiers (HO = hyperparameter optimisation, FS = feature selection, obj = optimisation objective).

with higher precision than in cross-validation, but lower recall. Interestingly, both $F_1$-optimised systems obtain better recall on the held-out test set than their $F_2$-optimised counterparts.

Figure 4 presents learning curves for each system. Training error with SVM is very small with training scores around the maximum for all systems. The cross-validation score keeps increasing as more training data is added, which suggests that obtaining additional data would be beneficial. This is true in particular for the severity task, which has fewer support instances and shows a threefold increase in validation score as training size increases. Held-out scores improve with additional data as well. For $F_2$-optimized systems, however, the learning effect is less outspoken: validation scores level off more, and held-out score decreases for relevance and is erratic for severity. We believe the variance of these systems is too high. Overfitting would also explain why the $F_2$-optimised systems achieve lower recall on the held-out test set than the systems optimized for $F_1$.

### *6.6. Scaling and qualitative analysis*

For the cross-validation experiments, we reported results on a dataset with a high incidence of suicide-related material. However, the incidence of positive instances in real-world user generated content is much lower. We are not aware of any studies that estimate the ratio of suicide-related messages in social media, but given the low epidemiological ratios and the assumption that an individual's suicidal behaviour will not always be manifest in his or her social media activity,

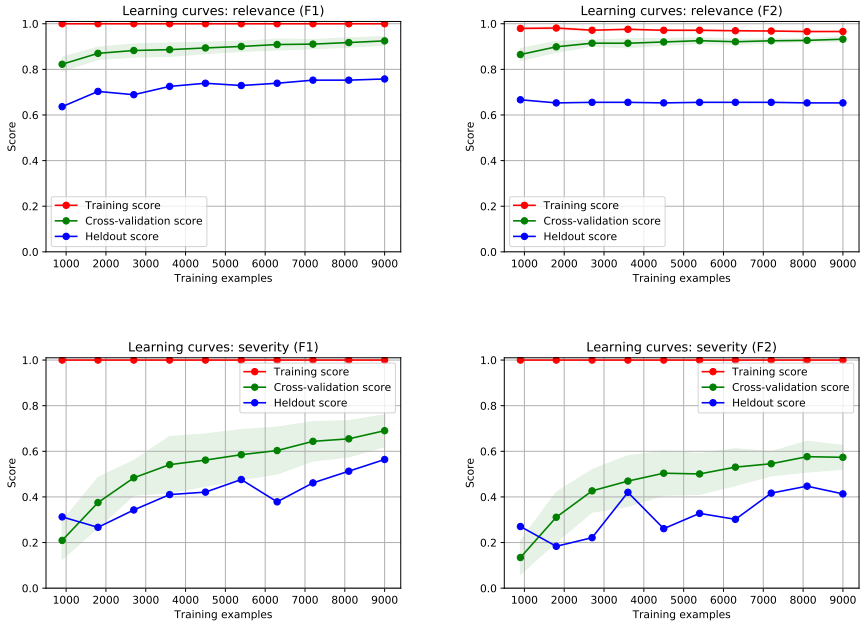


Figure 4: Training, cross-validation and held-out scores for four selected classifiers (top: relevance, bottom: severity), as a function of the available amount of training examples. Reported scores are $F_1$ on the left and $F_2$, in line with the optimisation objective of the classifiers.

we can assume the ratio to be very low. A classifier that is not able to find this needle in a haystack, but which flags many irrelevant posts for review, might present prevention workers with an overwhelming amount of noise. In order to determine to what extent the trained models are capable of separating suicidal posts from a vast pool of unrelated material, and thus to get an impression of their practical usability, we performed a set of scaling experiments. We sampled increasingly large subsets (of 20 000, 70 000 and 200 000 posts) from the same corpus the experimental corpus was derived from. The held-out test set of 10 040 posts, where 58 posts are suicide-related and 26 of those contain a severe risk of suicide, was incrementally enlarged with these subsets, leading to an increasingly smaller ratio of known positive instances in the data. Since the scaling datasets have not been annotated, we do not know if they contain suicide-related posts. We therefore cannot report on the recall of our models,

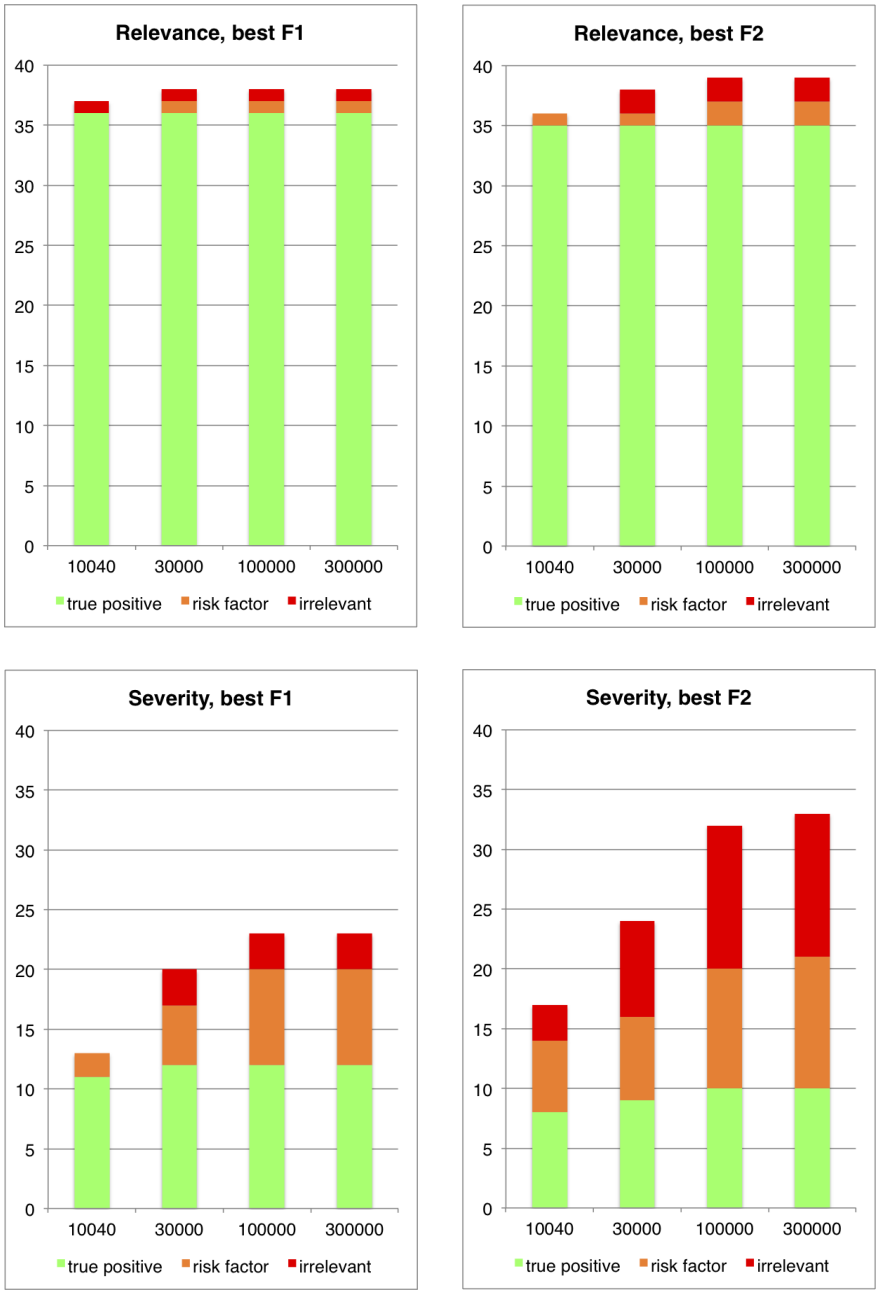


Figure 5: Number of true and false positives on the held-out and scaling datasets, using the best $F_1$, and $F_2$ classifiers for the relevance and severity tasks. False positives are divided into two groups, depending on whether they contain risk factors.

but the scaling experiments shed light on their usability in terms of precision: as dataset size increases, how many false positives (i.e. noise) are added to the small number of known true positives?

When applying the four selected classifiers on the four scaling corpora, we obtain stable predictions for the known positives on each dataset, but we are mainly interested in seeing how the number of false positives increases with data set size. Figure 5 presents the absolute number of true and false positives flagged by the system in the four scaling corpora, for each task and optimisation objective. The positive predictions were manually evaluated and classified as either relevant (true positives from inside or outside of the 58 (26) known positive sample, green), not relevant but containing risk factors (borderline cases, orange), and irrelevant (false positives, red). The results show that the relevance classifiers are able to keep the amount of noise minimal, even on the full 300 000 post corpus. When the system is scaled to large datasets with high class skew, it retains very high precision: false positives are virtually absent. The system is generally conservative in its predictions. A qualitative analysis of the false negatives (from the 40 post positive sample) reveals that they lack explicit mentions of suicide, suggesting that in order to improve recall, more implicit references need to be detected.

The scaling results of the severity classifiers indicate that they make many positive predictions outside the known positive sample, unlike the relevance classifiers. The qualitative analysis on the scaling dataset reveals that the severity models are most successful in detecting posts in which an author personally discloses suicide ideation, especially when this is done in explicit terms. Posts about a third person are often incorrectly dismissed as relevant but insevere, possibly because of confusion with posts about celebrities committing suicide, which always receive a *relevant but insevere* annotation. More false positives are produced on this big dataset than with the relevance system, although more than half of them contain suicide risk factors, and are therefore not entirely irrelevant. At less than 0.01% of the data, noise is still acceptably low for the system to be usable in a real-world application.

## 7. Conclusions and future work

The current study presents the first investigation of automatic text classification as a solution for detecting suicidality in the (online) population. Contrary to previous studies, it does not rely solely on keyword filtering to find suicide-related messages, but makes use of machine learning to improve performance in terms of precision and recall, which can both be evaluated with the manually annotated experimental corpus.

Experiments focused on two tasks: the detection of suicide-related posts, and of severe, high-risk content. Results show that both types of messages can be detected with high precision. Therefore, the amount of noise generated by the system is minimal, even on very large datasets, making it usable in a real-world prevention setting. Recall is high for the relevance task, but at around 60%, it is considerably lower for severity. This is mainly attributable to implicit references to suicide, which often go undetected.

To improve classification performance, the models were optimised using feature selection, hyperparameter optimisation, or a combination of both. A distributed genetic algorithm approach proved successful in finding good solutions for this complex search problem, and resulted in better models. After feature selection, a variety of information sources was found to be informative for both tasks, including token and character ngram bags-of-words, features based on LSA topic models, polarity lexicons and named entity recognition, and suicide-related terms extracted from a background corpus. The results indicate that it is beneficial to abstract away from the surface word forms, given the success of topic model and character ngram features.

An important limitation to using supervised text classification for suicide prevention is the dependence on labeled data. Although suicide-related data that has been annotated by experts is very valuable, it is problematic to obtain in at least two respects: the very low incidence in general-domain data makes manual annotation prohibitively slow and expensive, and collecting and distributing data from suicide prevention centers is complex or potentially un-

desirable for reasons of privacy and consent. Furthermore, the data and system presented in this paper is specific to Dutch. In future work, we intend to investigate cross-lingual transfer as a method to address these limitations: it could allow to build systems for other languages, based on the Dutch training data, and to pool the resources available for different languages into a single larger and more diverse training set. Cross-lingual transfer has been shown to improve performance for resource-poor languages, for tasks including POS tagging, dependency parsing and named entity recognition. Typical approaches use bitext [48, 43], although recent work reduced [40] or entirely eliminated the dependence on parallel corpora [49].

The systems will also be evaluated in a real-world prevention setting. The responsiveness of forum moderators to suicidal posts will be compared in a setup with and without the software. Another alley for future work is to reduce the linguistic noise that is typical of user-generated content. Automatic text normalisation techniques may be applied to bring text closer to the linguistic norm and reduce variation. By improving lexical recall, overall performance could be improved.

## Acknowledgments

The first author is a Postdoctoral Fellow of the Research Foundation - Flanders. This study was supported by the BOF project SubTLe (code HGA07J0313T) and the IWT SBO project AMiCA (code 120007). We acknowledge the annotation efforts by staff and volunteers at the Flemish Suicide Prevention Center.

## References

[1] Baroni, M., Bernardini, S., 2004. BootCaT : Bootstrapping Corpora and Terms from the Web. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC '04). Lisbon, Portugal, pp. 1313–1316.

[2] Carletta, J., 1996. Assessing Agreement on Classification Tasks: The Kappa Statistic. Computational Linguistics 22 (2), 249–254.

[3] Chang, C.-C., Lin, C.-J., 2011. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2 (3), 1–27.

[4] Cortez, P., Peralta, J., 2014. Global and decomposition evolutionary support vector machine approaches for time series forecasting. Neural Computing and Applications 25 (5), 1053–1062.

[5] De Smedt, T., Daelemans, W., 2012. Pattern for Python. Journal of Machine Learning Research 13, 2063–2067.

[6] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6 (2), 182–197.

[7] Desmet, B., Hoste, V., Verstraeten, D., Verhasselt, J., 2013. Gallop Documentation. Tech. rep.

[8] Forman, G., 2003. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. Journal of Machine Learning Research 3, 1289–1305.

[9] Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., Gagné, C., 2012. DEAP: Evolutionary Algorithms Made Easy. Journal of Machine Learning Research 13, 2171–2175.

[10] Gisle, L., 2008. Mentale Gezondheid. Tech. rep., Wetenschappelijk Instituut Volksgezondheid, Brussel.

[11] Goldberg, D. E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley.

[12] Goldberg, D. E., Deb, K., 1991. A Comparative Analysis of Selection Schemes used in Genetic Algorithms. In: Foundations of Genetic Algorithms. Vol. 51. Morgan Kaufmann Publishers, San Mateo, CA, USA, pp. 69–93.

[13] Goldberg, Y., Elhadad, M., 2008. splitSVM: Fast, Space-Efficient, non-Heuristic, Polynomial Kernel Computation for NLP Applications. In: Proceedings of ACL-08: HLT, Short Papers. No. June. Association for Computational Linguistics, Columbus, US, pp. 237–240.

[14] Haerian, K., Salmasian, H., Friedman, C., 2012. Methods for Identifying Suicide or Suicidal Ideation in EHRs. In: AMIA Annual Symposium Proceedings. pp. 1244–1253.

[15] Holland, J. H., 1975. Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence. MIT Press.

[16] Hoste, V., 2005. Optimization Issues in Machine Learning of Coreference Resolution. Ph.D. thesis, Universiteit Antwerpen.

[17] Hsu, C.-w., Chang, C.-c., Lin, C.-j., 2010. A Practical Guide to Support Vector Classification 1 (1), 1–16.

[18] Huang, Y.-P., Goh, T., Liew, C. L., Dec. 2007. Hunting Suicide Notes in Web 2.0 - Preliminary Findings. In: Ninth IEEE International Symposium on Multimedia Workshops (ISMW 2007). IEEE, pp. 517–521.

[19] Jashinsky, J., Burton, S. H., Hanson, C. L., West, J., Giraud-Carrier, C., Barnes, M. D., Argyle, T., Oct. 2014. Tracking Suicide Risk Factors Through Twitter in the US. Crisis 35 (1), 51–59.

[20] Jeni, L. A., Cohn, J. F., Torre, F. D. L., 2013. Facing Imbalanced Data: Recommendations for the Use of Performance Metrics. In: Proceedings of the 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction.

[21] Jijkoun, V., Hofmann, K., 2009. Generating a Non-English Subjectivity Lexicon : Relations That Matter. In: Proceedings of the 12th Conference of the European Chapter of the ACL. No. April. Association for Computational Linguistics, Athens, Greece, pp. 398–405.

[22] Joachims, T., 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Machine Learning: ECML-98, 137–142.

[23] Kerkhof, A., van Luyn, J., 2010. Suïcidepreventie in de praktijk. Bohn Stafleu van Loghum, Houten.

[24] Kökciyan, N., Çelebi, A., Özgür, A., Üsküdarli, S., 2013. BOUNCE: Sentiment Classification in Twitter using Rich Feature Sets. In: Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013). Atlanta, Georgia, USA, pp. 554–561.

[25] Leenaars, A. A., 1988. Suicide notes: Predictive clues and patterns. Human Sciences Press, New York.

[26] Macken, L., Lefever, E., Hoste, V., 2013. TExSIS: Bilingual Terminology Extraction from Parallel Corpora Using Chunk-based Alignment. Terminology 19 (1), 1–30.

[27] Matykiewicz, P., Duch, W., Pestian, J., 2009. Clustering semantic spaces of suicide notes and newsgroups articles (June), 179–184.

[28] Mendes, P. N., Jakob, M., García-Silva, A., Bizer, C., 2011. DBpedia Spotlight : Shedding Light on the Web of Documents. In: Proceedings of the 7th International Conference on Semantic Systems. pp. 1–8.

[29] O'Dea, B., Wan, S., Batterham, P. J., Calear, A. L., Paris, C., Christensen, H., 2015. Detecting suicidality on Twitter. Internet Interventions 2 (2), 183–188.

URL http://linkinghub.elsevier.com/retrieve/pii/S2214782915000160

[30] Pestian, J., Matykiewicz, P., Linn-Gust, M., South, B., Uzuner, O., Wiebe, J., Cohen, K. B., Hurdle, J., Brew, C., Jan. 2012. Sentiment Analysis of Suicide Notes: A Shared Task. Biomedical Informatics Insights 5, 3–16.

[31] Pestian, J., Nasrallah, H., Matykiewicz, P., Bennett, A., Leenaars, A., 2010. Suicide Note Classification Using Natural Language Processing : A Content Analysis. Biomedical Informatics Insights 3, 19–28.

[32] Peter, J., Valkenburg, P. M., Schouten, A. P., Oct. 2005. Developing a model of adolescent friendship formation on the internet. Cyberpsychology & behavior 8 (5), 423–30.

[33] Rehurek, R., Sojka, P., 2010. Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. Valletta, Malta, pp. 45–50.

[34] Saito, T., Rehmsmeier, M., 2015. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. PLoS ONE 10 (3), 1–21.

[35] Salzberg, S. L., 1997. On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach. Data Mining and Knowledge Discovery 1 (3), 317–327.

[36] Sebastiani, F., 2002. Machine Learning in Automated Text Categorization. ACM computing surveys (CSUR) 34 (1), 1–47.

[37] Shapero, J. J., 2011. The Language of Suicide Notes. Ph.D. thesis.

[38] Shneidman, E. S., Farberow, N. L., 1957. Clues to Suicide. Vol. 71. McGraw-Hill, New York.

[39] Stenetorp, P., Pyysalo, S., Topic, G., Ohta, T., Ananiadou, S., Tsujii, J., 2012. BRAT: a web-based tool for NLP-assisted text annotation. In: Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL). Avignon, France, pp. 102–107.

[40] Täckström, O., McDonald, R., Uszkoreit, J., 2012. Cross-lingual Word Clusters for Direct Transfer of Linguistic Structure. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT). pp. 477–487.

[41] Tran, T., Luo, W., Phung, D., Harvey, R., Berk, M., Kennedy, R. L., Venkatesh, S., Jan. 2014. Risk stratification using data from electronic medical records better predicts suicide risks than clinician assessments. BMC Psychiatry 14 (76).

[42] Värnik, P., Mar. 2012. Suicide in the World. International Journal of Environmental Research and Public Health 9 (3), 760–71.

[43] Wang, M., Manning, C. D., 2014. Cross-lingual Projected Expectation Regularization for Weakly Supervised Learning. Transactions of the Association for Computational Linguistics 2, 55–66.

[44] Whitley, D., 1994. A genetic algorithm tutorial. Statistics and Computing 4, 65–85.

[45] World Health Organization, 2011. Causes of Death 2008 Summary Tables. Tech. rep., Health Statistics and Informatics Department, Geneva, Switzerland.

[46] Yang, Y., Liu, X., 1999. A Re-examination of Text Categorization Methods. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, pp. 42–49.

[47] Yang, Y., Pedersen, J. O., 1997. A comparative study on feature selection in text categorization. In: Proceedings of the 14th International Conference on Machine Learning (ICML). pp. 412–420.

[48] Yarowsky, D., Ngai, G., Wicentowski, R., 2001. Inducing Multilingual Text Analysis Tools via Robust Projection across Aligned Corpora. In: Proceedings of the first international conference on Human Language Technology research. pp. 1–8.

[49] Zirikly, A., Hagiwara, M., 2015. Cross-lingual Transfer of Named Entity Recognizers without Parallel Corpora. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. Association for Computational Linguistics, Beijing, China, pp. 390–396.