Recognising suicidal messages in Dutch social media

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

Abstract

Early detection of suicidal thoughts is an important part of effective suicide prevention. Such thoughts may be expressed online, especially by young people. This paper presents on-going work on the automatic recognition of suicidal messages in social media. We present experiments for automatically detecting relevant messages (with suicide-related content), and those containing suicide threats. A sample of 1357 texts was annotated in a corpus of 2674 blog posts and forum messages from Netlog, indicating relevance, origin, severity of suicide threat and risks as well as protective factors. For the classification experiments, Naive Bayes, SVM and KNN algorithms are combined with shallow features, i.e. bag-of-words of word, lemma and character ngrams, and post length. The best relevance classification is achieved by using SVM with post length, lemma and character ngrams, resulting in an F-score of 85.6% (78.7% precision and 93.8% recall). For the second task (threat detection), a cascaded setup which first filters out irrelevant messages with SVM, and then predicts the severity with KNN, performs best: 59.2% F-score (69.5% precision and 51.6% recall).

Keywords: suicide prevention, suicide detection, social media

1. Introduction

Suicide is a major public health concern worldwide, particularly among young people. In 2010, it was the third leading cause of death in US citizens aged between 1 and 44 years (Miniño and Murphy, 2012). Successful prevention hinges on early risk recognition and referral to appropriate support. Social media are increasingly becoming an outlet for suicidal thoughts, which suicide prevention stakeholders are keen not to ignore. Considering the volume of text produced in such media, manual monitoring is practically unfeasible. An automatic procedure for filtering out alarming messages would allow them to find and quickly react to suicide ideation or incitement online. The effectiveness of such short interventions has been proven (Christensen et al., 2004).

Computational linguistics research on topics concerning suicide is fairly recent. Shapero (2011) offers a detailed analysis of the language in fake and genuine suicide notes, and Pestian et al. (2010) investigate whether they can be automatically distinguished for forensic or clinical purposes. A shared task on automatic emotion detection in suicide notes was organised in 2011 (Pestian et al., 2012). The geographic correlation between suicide mortality rates and the occurrence of risk factors in tweets is described in Jashinsky et al. (2013), along with potential implications for online suicide prevention. To our knowledge, no prior work exists on the annotation and recognition of potentially alarming messages in social media, that could be relevant for suicide prevention centers or website administrators.

In this paper, we introduce a cascaded annotation strategy for recognising suicidal content, describe the collection and annotation of an evaluation corpus, and present experiments on automatic recognition.

2. Annotation of suicide risk

Suicide-related text can present itself in many forms, and not all of it is relevant for prevention purposes. In order to develop an annotation scheme that was motivated by practice, we collaborated with the Belgian Centre for Suicide Prevention (CPZ[1]). This resulted in a cascaded scheme where the depth of annotation depends on whether a text (which in social media terms could be a blog post, forum message, tweet, etc.) matches certain criteria. It allows to derive multiple classifications for various applications.

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 *origin* 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 suicide content is determined as either the author, some other person, or both. Incitement to commit suicide is flagged.

The *severity* of the suicide threat is annotated, depending on the presence of suicide thoughts or plans, and the *language* used to describe them.

For all text types, the presence of *risk factors* and *protective factors* are indicated. Risk factors are trivializations, motivations or methods for suicide. The detection of risk factors is not only relevant in text that is written personally, but in other text genres as well. Copycat behaviour is known to occur when risk factors are included in journalistic articles, for example. Protective factors are referrals to counsel, such as the CPZ emergency line.

The above features are annotated at the document level. Five types of *text spans* may also be marked in the text itself: risk and protective factors, citations, and passages that are alarming (e.g. *This bullying needs to stop*) or clearly suicidal (e.g. *I want to sleep forever*). Such spans may pro-

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

vide valuable lexical evidence of what makes a text alarming.

3. Corpus

Two corpora of blog posts and forum messages were collected from the Dutch section of Netlog[2], a social networking site that is popular amongst teenagers in particular. The first was collected using four keywords: *suicide* and its Dutch translations *suïcide, zelfdoding* and *zelfmoord*. This yielded 1,380 documents, most of which were suicide-related. The second corpus contained 373,349 documents that were selected randomly.

The annotation scheme described in Section 2. was implemented in the online annotation tool brat (Stenetorp et al., 2012). Some modifications were made to allow text-level annotation. A team of trained crisis responders at CPZ were tasked with annotating the keyword-selected corpus whenever they had the time, which resulted in 1357 annotated texts.

Of the 1357 texts that were annotated, 1024 were found to be relevant (with content matching the clinical definition of suicide). Of those, 221 presented a severe suicide risk.

A section of the reference corpus (1317 texts) was also screened for suicidal content, but none was found. This reference corpus was combined with the annotated corpus, to form the experimental corpus of 2674 texts.

4. Experiments on automatic detection

The task of automatically detecting suicide-related content can be defined in different ways, depending on the application. In a broad sense, prevention workers may be interested in finding all suicide-related content, e.g. to monitor for the presence of risk factors. With limited resources however, it may be preferable to only filter out personal messages that indicate a severe suicide risk and require immediate attention. We present experiments on both tasks.

4.1. Experimental setup

The experimental corpus, a combination of an annotated corpus and a reference corpus, provided 2674 texts for classification (see 3.). For the relevance filtering task, this gave a split of 1024 positive versus 1650 negative instances (ratio ~1:1.6), the severity task had 221 positive versus 2453 negative instances (ratio ~1:11). In reality, the proportions would be skewed much more towards the negative classes.

Performance was measured with F-score on the minority positive class. Because of the data skewness, measures such as accuracy would favour negative classification. F-score with a standard β of 1 was used to ensure a harmonic mean between precision and recall. For our tasks, both are expected to have equal importance: to find what needs to be found, but not flood the user with false positives. In cases where recall is of particular importance (e.g. for cascaded classifiers, see below), we also discuss F-scores with β=2, such that recall has twice the weight of precision in the F-score calculation.

We used Pattern, a Python package for web mining, NLP and machine learning (De Smedt and Daelemans, 2012) to experiment with 3 classification algorithms: Naive Bayes (NB, with Bernoulli or multinomial distributions and alpha smoothing values between 10^{-6} and 10^{-1}), Support Vector Machines (SVM, with linear or polynomial kernels and cost values between 10^{-5} and 10^{2}) and k-nearest neighbour (KNN, cosine or hamming distance metric and k=5 or 10).

For our experiments, we tested whether suicidal content could be detected through shallow lexical markers in text. Various features were derived from the texts:

- Word unigram and bigram bags-of-words
- Lemma unigram bag-of-words, to provide some abstraction from the word level and somewhat decrease vector sparseness
- Character bigram, trigram and fourgram bags-of-words, again to provide abstraction, because the noisy nature of social media content severely hurts the accuracy of lemmatization.
- Post length, expressed as the logarithm of the number of characters. This feature was included to correct classifier behaviour where very short posts are incorrectly tagged as relevant.

Using gridsearches, the best classifier-parameter pairings were determined for a number of feature combinations. Evaluation was done using 3-fold cross-validation.

We used these features and classifiers as a one-shot approach for both the relevance and severity tasks. For the severity task, we also experiment with cascaded classifiers, where a first classifier filters out irrelevant messages, and a second one predicts severity on the remaining instances.

We compare the results to two baseline systems. The first system always predicts the positive class, with perfect recall but low precision. This baseline system scores 55.4% and 15.3% F-score on the first and second task, respectively. The second baseline labels a post as positive if one of the four keywords used to collect the corpus are present. This baseline scores 77.2% on the first task, and 27.5% F-score on the second.

4.2. Results and discussion

4.2.1. Relevance task

The results obtained for the relevance task, with varying algorithms and feature combinations, are presented in Tables 1 and 2.

The scores show that SVM consistently outperforms KNN, which in turn outperforms NB. NB performs particularly poorly, with many scores under the positive baseline. Using only words or lemmas, its results approach the keyword baseline, but never outperforms it. SVM is always better than baseline, except when word bigrams are used.

Both SVM and KNN benefit from character trigrams and fourgrams, which are selected for most well-performing classifiers. Post length is also useful for relevance filtering. Words or lemmas are beneficial too, when combined with character ngrams.

Performance goes up with the length of character ngrams, and the abstraction they offer is useful, as evidenced by the

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

Relevance	**NB** (distribution, α)	**SVM** (kernel, C)	**KNN** (distance, k)
w	***71.1*** **bernoulli**, 10^{-2}	80.6 linear, 10^{-3}	*76.1* cosine, 1
w2	*46.7* bernoulli, 10^{-1}	*68.8* linear, 10^{-2}	*75.3* cosine, 1
w, w2	*42.4* bernoulli, 10^{-1}	80.9 linear, 10^{-3}	*75.7* cosine, 1
l	***71.1*** **bernoulli**, 10^{-2}	81.2 linear, 10^{-2}	77.6 cosine, 1
ch2	*60.7* bernoulli, 10^{-6}	80.8 linear, 10^{-4}	*75.3* cosine, 1
ch3	*45.8* bernoulli, 10^{-3}	82.8 linear, 10^{-4}	*77.2* cosine, 1
ch4	*27.8* bernoulli, 10^{-1}	82.5 linear, 10^{-3}	79.0 cosine, 1
ch3, ch4	*11.6* bernoulli, 10^{-1}	83.4 linear, 10^{-4}	77.8 cosine, 1
w, ch2	*57.0* bernoulli, 10^{-2}	81.1 linear, 10^{-4}	*74.9* cosine, 1
w, ch3	*39.2* bernoulli, 10^{-2}	84.0 linear, 10^{-4}	*76.9* cosine, 1
w, ch4	*23.5* bernoulli, 10^{-1}	83.9 linear, 10^{-4}	79.3 cosine, 1
w, ch2, ch3	*32.4* bernoulli, 10^{-4}	82.3 linear, 10^{-4}	*75.3* cosine, 1
w, ch3, ch4	*9.7* bernoulli, 10^{-1}	84.4 linear, 10^{-4}	77.6 cosine, 1
w, ch2, ch3, ch4	*8.4* bernoulli, 10^{-1}	82.7 linear, 10^{-4}	*74.9* cosine, 1
l, ch2	*56.4* bernoulli, 10^{-1}	80.5 linear, 10^{-4}	*74.8* cosine, 1
l, ch3	*38.9* bernoulli, 10^{-2}	82.7 linear, 10^{-4}	77.9 cosine, 1
l, ch4	*23.1* bernoulli, 10^{-1}	82.9 linear, 10^{-3}	80.0 cosine, 1
l, ch2, ch3	*31.8* bernoulli, 10^{-1}	82.1 linear, 10^{-4}	*75.2* cosine, 1
l, ch3, ch4	*10.0* bernoulli, 10^{-1}	82.9 linear, 10^{-4}	78.0 cosine, 1
l, ch2, ch3, ch4	*8.3* bernoulli, 10^{-1}	82.4 linear, 10^{-4}	*75.5* cosine, 1
l, ch2, pl	*56.3* bernoulli, 10^{-1}	83.1 linear, 10^{-5}	79.0 cosine, 1
l, ch3, pl	*39.2* bernoulli, 10^{-2}	82.9 linear, 10^{-5}	**82.1 cosine, 1**
l, ch4, pl	*22.5* bernoulli, 10^{-1}	83.5 linear, 10^{-5}	**83.2 cosine, 1**
l, ch2, ch3, pl	*31.7* bernoulli, 10^{-3}	84.6 linear, 10^{-5}	79.0 cosine, 1
l, ch3, ch4, pl	*10.0* bernoulli, 10^{-1}	**84.8 linear,** 10^{-5}	81.5 cosine, 1
l, ch2, ch3, ch4, pl	*8.2* bernoulli, 10^{-2}	**85.6 linear,** 10^{-5}	78.6 cosine, 1
positive baseline	55.4		
keyword baseline	77.2		

Table 1: Results for the relevance filtering task, reported as 3-fold cross-validated F-score on the positive class. Feature sets can contain words (w), word bigrams (w2), lemmas (l), character ngrams (ch2, ch3, ch4) and post length (pl). For each feature set, the score and settings of the best classifier (after a hyperparameter gridsearch) is given for NB, SVM and KNN. Scores in italics are below the keyword baseline, the 2 strongest classifiers of each type are boldfaced.

	Precision	Recall	F (β=1)	F (β=2)
Positive baseline	38.3	100.0	55.4	75.6
Keyword baseline	81.7	73.0	77.2	74.6
SVM (l, ch3, ch4, pl)	78.6	92.1	84.8	89.0
SVM (l, ch2, ch3, ch4, pl)	78.7	93.8	85.6	90.4
KNN (l, ch3, pl)	73.5	92.9	82.1	88.2
KNN (l, ch4, pl)	77.4	89.9	83.2	87.1

Table 2: Precision, recall and F-scores (with β 1 and 2) for selected relevance classifiers (boldfaced in Table 1).

lower scores with words or lemmas versus character fourgrams.

The best classifier, SVM with lemmas, all character ngrams and post length, achieves an F-score of 85.6%, with precision at 78.7% and recall at 93.8%. This means that ~6% of suicidal messages are missed, and the suggested positives contain ~20% noise. This is an improvement over the keyword baseline, which would fail to retrieve 27% of the messages, at the same level of noise. This improvement is reflected best in the F-scores with β=2, which doubles the importance of recall of suicidal messages.

4.2.2. Severity task: one-shot

Tables 3 and 4 present the scores for the one-shot severity task. In this approach, a classifier needs to detect the presence of a suicide threat without filtering out irrelevant messages first.

As with the relevance task, NB struggles to beat the baseline, and is increasingly confused as the number of features goes up. Only word or lemma bag-of-words perform reasonably well.

We can make a number of observations from the results. Both SVM and KNN perform well, yielding promising scores in comparison to the baseline.

It is interesting to note that whereas KNN prefers longer character ngrams, SVM prefers shorter ones, in contrast with the findings for the previous task, which could be considered more of a topic detection task. The best combinations include lemma bag-of-words and post length as well.

The best KNN classifier finds 111 out of 221 positive instances (recall 50.2%), with 77 false positives (precision 59.0%). The keyword baseline sacrifices precision to obtain better recall, finding 156 true positives at the expense

Severity	**NB** (distribution, α)		**SVM** (kernel, C)		**KNN** (distance, k)	
w	**40.9**	**bernoulli**, 10^{-5}	35.6	linear, 10^{-2}	48.1	cosine, 1
w2	*13.9*	bernoulli, 10^{-1}	*14.8*	linear, 10^{-2}	*24.2*	cosine, 1
w, w2	*12.5*	bernoulli, 10^{-4}	35.3	linear, 10^{-2}	44.5	cosine, 1
l	**41.1**	**bernoulli**, 10^{-5}	41.9	linear, 10^{-2}	47.4	cosine, 1
ch2	*25.9*	bernoulli, 10^{-6}	45.5	linear, 10^{-3}	39.7	cosine, 1
ch3	*27.3*	bernoulli, 10^{-6}	43.7	linear, 10^{-3}	47.8	cosine, 1
ch4	*14.1*	bernoulli, 10^{-5}	44.1	linear, 10^{-3}	51.1	cosine, 1
ch3, ch4	*4.8*	bernoulli, 10^{-3}	46.3	linear, 10^{-3}	51.4	cosine, 1
w, ch2	32.4	bernoulli, 10^{-6}	46.9	linear, 10^{-3}	36.7	cosine, 1
w, ch3	*21.3*	bernoulli, 10^{-5}	44.8	linear, 10^{-3}	49.8	cosine, 1
w, ch4	*11.7*	bernoulli, 10^{-3}	47.9	linear, 10^{-3}	52.2	cosine, 1
w, ch2, ch3	*15.3*	bernoulli, 10^{-6}	47.5	linear, 10^{-3}	41.3	cosine, 1
w, ch3, ch4	*3.4*	bernoulli, 10^{-2}	46.5	linear, 10^{-3}	51.0	cosine, 1
w, ch2, ch3, ch4	*2.8*	bernoulli, 10^{-3}	48.0	linear, 10^{-3}	42.6	cosine, 1
l, ch2	31.9	bernoulli, 10^{-6}	48.9	linear, 10^{-2}	38.8	cosine, 1
l, ch3	*21.9*	bernoulli, 10^{-6}	45.7	linear, 10^{-3}	49.6	cosine, 1
l, ch4	*10.8*	bernoulli, 10^{-3}	45.0	linear, 10^{-2}	52.0	cosine, 1
l, ch2, ch3	*17.0*	bernoulli, 10^{-6}	44.0	linear, 10^{-2}	46.2	cosine, 1
l, ch3, ch4	*3.3*	bernoulli, 10^{-1}	45.5	linear, 10^{-3}	52.1	cosine, 1
l, ch2, ch3, ch4	*2.9*	bernoulli, 10^{-2}	44.9	linear, 10^{-3}	43.8	cosine, 1
l, ch2, pl	31.8	bernoulli, 10^{-6}	**49.4**	**linear,** 10^{-2}	41.3	cosine, 1
l, ch3, pl	*21.7*	bernoulli, 10^{-6}	47.8	linear, 10^{-2}	53.1	cosine, 1
l, ch4, pl	*11.1*	bernoulli, 10^{-3}	48.3	linear, 10^{-2}	**53.5**	**cosine, 1**
l, ch2, ch3, pl	*16.8*	bernoulli, 10^{-6}	**49.0**	**linear,** 10^{-3}	43.6	cosine, 1
l, ch3, ch4, pl	*3.4*	bernoulli, 10^{-3}	47.1	linear, 10^{-3}	**54.3**	**cosine, 1**
l, ch2, ch3, ch4, pl	*3.2*	bernoulli, 10^{-4}	48.3	linear, 10^{-3}	43.5	cosine, 1
positive baseline	15.3					
keyword baseline	27.5					

Table 3: Results for the one-shot severity filtering task, reported as 3-fold cross-validated F-score on the positive class. Feature sets can contain words (w), word bigrams (w2), lemmas (l), character ngrams (ch2, ch3, ch4) and post length (pl). For each feature set, the score and settings of the best classifier (after a hyperparameter gridsearch) is given for NB, SVM and KNN. Scores in italics are below the keyword baseline, the 2 strongest classifiers of each type are boldfaced.

	Precision	Recall	F (β=1)	F (β=2)
Positive baseline	8.3	100.0	15.3	31.1
Keyword baseline	17.0	70.6	27.5	43.4
SVM (l, ch2, pl)	50.5	48.4	49.4	48.8
SVM (l, ch2, ch3, pl)	57.7	42.5	49.0	44.9
KNN (l, ch4, pl)	60.6	48.0	53.5	50.0
KNN (l, ch3, ch4, pl)	59.0	50.2	54.3	51.8

Table 4: Precision, recall and F-scores (with β 1 and 2) for selected one-shot severity classifiers (boldfaced in Table 3).

of 700 extra false negatives.

4.2.3. Severity task: cascaded

A possible drawback of the one-shot approach is that a classifier needs to make multiple decisions to arrive at the subset of alarming suicide-related messages. We therefore also experimented with severity classifiers that work on the output of a relevance filter.

As a filter, we used the two relevance classifiers with the highest F(β=2)-score, because precision mistakes can be addressed in the second step of the cascade, whereas recall mistakes cannot, because false negatives are not fed to the second classifier. These were both SVM classifiers.

As a cascaded severity classifier, we used KNN and SVM with all feature sets and hyperparameters (Table 5).

The cascaded approach improves the best F-scores for both SVM and KNN, by 2.7 and 4.9 percentage points, respectively. KNN still benefits from a large number of features, whereas SVM works best when it uses character fourgrams only.

The benefit of using a cascade is primarily improved precision, as can be seen in Table 6. A filter weeds out potential false positives, but does not help much to improve recall. The second classifier can be better tuned to the severity task however, which could provide better recall as well.

In our experiments, we observe significant gains in precision, and minor improvements in recall. The best cascaded KNN classifier finds 114 messages containing a suicide threat (three more than the best one-shot classifier), and produces 50 false positives (as opposed to 77 with one-shot classification).

Relevance filter	SVM (l, ch3, ch4, pl)				SVM (l, ch2, ch3, ch4, pl)			
+ cascaded severity	SVM (kernel, C)		KNN (distance, k)		SVM (kernel, C)		KNN (distance, k)	
w	47.8	linear, 10^{-2}	48.2	cosine, 1	47.8	linear, 10^{-2}	47.9	cosine, 1
w2	39.4	linear, 10^{-2}	31.1	cosine, 1	39.3	linear, 10^{-2}	31.0	cosine, 1
w, w2	46.8	linear, 10^{-2}	45.7	cosine, 1	46.9	linear, 10^{-2}	45.0	cosine, 1
l	51.6	linear, 10^{-2}	53.2	cosine, 1	51.4	linear, 10^{-2}	53.5	cosine, 1
ch2	50.3	linear, 10^{-3}	43.4	cosine, 1	50.0	linear, 10^{-3}	43.6	cosine, 1
ch3	49.2	linear, 10^{-3}	52.8	cosine, 1	49.3	linear, 10^{-3}	53.0	cosine, 1
ch4	**52.3**	**linear,** 10^{-3}	52.8	cosine, 1	**52.1**	**linear,** 10^{-3}	52.9	cosine, 1
ch3, ch4	48.8	linear, 10^{-3}	54.2	cosine, 1	49.1	linear, 10^{-3}	54.5	cosine, 1
w, ch2	51.4	linear, 10^{-2}	41.4	cosine, 1	51.7	linear, 10^{-2}	41.3	cosine, 1
w, ch3	51.1	linear, 10^{-3}	53.9	cosine, 1	51.2	linear, 10^{-3}	53.9	cosine, 1
w, ch4	**52.1**	**linear,** 10^{-3}	55.8	cosine, 1	**51.8**	**linear,** 10^{-3}	55.6	cosine, 1
w, ch2, ch3	48.4	linear, 10^{-1}	45.0	cosine, 1	47.9	linear, 10^{-1}	44.8	cosine, 1
w, ch3, ch4	50.0	linear, 10^{-3}	55.7	cosine, 1	50.3	linear, 10^{-3}	56.0	cosine, 1
w, ch2, ch3, ch4	48.5	linear, 10^{-3}	44.0	cosine, 1	48.2	linear, 10^{-3}	43.7	cosine, 1
l, ch2	51.6	linear, 10^{-2}	42.8	cosine, 1	51.6	linear, 10^{-2}	46.8	cosine, 1
l, ch3	48.9	linear, 10^{-3}	53.2	cosine, 1	49.3	linear, 10^{-3}	54.4	cosine, 1
l, ch4	49.9	linear, 10^{-3}	53.4	cosine, 1	49.7	linear, 10^{-3}	54.1	cosine, 1
l, ch2, ch3	49.0	linear, 10^{-3}	45.7	cosine, 1	48.8	linear, 10^{-3}	50.1	cosine, 1
l, ch3, ch4	50.1	linear, 10^{-3}	53.4	cosine, 1	50.5	linear, 10^{-3}	55.4	cosine, 1
l, ch2, ch3, ch4	48.7	linear, 10^{-3}	47.3	cosine, 1	48.6	linear, 10^{-3}	50.0	cosine, 1
l, ch2, pl	49.9	linear, 10^{-2}	41.6	cosine, 1	49.5	linear, 10^{-2}	41.4	cosine, 1
l, ch3, pl	49.9	linear, 10^{0}	55.6	cosine, 1	49.6	linear, 10^{-1}	55.8	cosine, 1
l, ch4, pl	48.2	linear, 10^{-2}	**57.0**	**cosine, 1**	48.4	linear, 10^{-2}	**57.7**	**cosine, 1**
l, ch2, ch3, pl	50.6	linear, 10^{-3}	45.3	cosine, 1	50.6	linear, 10^{-3}	45.7	cosine, 1
l, ch3, ch4, pl	49.7	linear, 10^{-1}	**58.9**	**cosine, 1**	49.9	linear, 10^{-1}	**59.2**	**cosine, 1**
l, ch2, ch3, ch4, pl	49.1	linear, 10^{-2}	46.9	cosine, 1	49.1	linear, 10^{-2}	47.2	cosine, 1
positive baseline	30.1				29.9			
keyword baseline	27.5				27.5			

Table 5: Results for the cascaded severity filtering task, reported as 3-fold cross-validated F-score on the positive class. Feature sets and hyperparameter settings are for the second classifier in the cascade. All scores are above the strongest baseline, the 2 strongest classifiers of each type are boldfaced.

	Precision	Recall	F (=1)	F (β=2)
Filter	SVM (l, ch3, ch4, pl)			
Positive baseline	17.9	95.5	30.1	51.1
Keyword baseline	17.0	70.6	27.5	43.4
SVM (ch4)	66.9	43.0	52.3	46.3
SVM (w, ch4)	67.1	42.5	52.1	45.9
KNN (l, ch4, pl)	66.7	49.8	57.0	52.4
KNN (l, ch3, ch4, pl)	69.3	51.1	58.9	54.0
Filter	SVM (l, ch2, ch3, ch4, pl)			
Positive baseline	17.7	96.4	29.9	51.0
Keyword baseline	17.0	70.6	27.5	43.4
SVM (ch4)	66.0	43.0	52.1	46.2
SVM (w, ch4)	66.2	42.5	51.8	45.8
KNN (l, ch4, pl)	67.7	50.2	57.7	53.0
KNN (l, ch3, ch4, pl)	69.5	51.6	59.2	54.4

Table 6: Precision, recall and F-scores (with β 1 and 2) for selected cascaded severity classifiers (boldfaced in Table 5).

4.2.4. General observations

Overall, SVM handles the inclusion of bad features best. Word bigrams, for example, degrade performance on all tasks, and have the least impact on SVM, likely because of its inherent feature selection. The problems were linearly separable, judging from the better results with linear over polynomial kernels. The optimal cost was around 10^{-4} for the first task, and 10^{-3} for the second. For KNN, cosine was the better distance metric, and a small neighbourhood was important: regardless of feature set, classifiers with k=1 outperformed those with k=5.

5. Conclusions and future work

This paper introduced an annotation scheme and corpus for the detection of various types of suicide-related content. The cascaded annotation approach may be of interest for other annotation tasks aimed at monitoring harmful online content, such as cyberbullying or sexual harassment.

The first classification experiments show that detecting posts on suicide is feasible (85.6% F-score), with SVM classifiers performing best with lemmas, character trigrams and fourgrams, and post length. Almost no relevant messages are missed, but with ~20% noise, the system would become increasingly flawed with larger reference corpora, where the number of suicidal messages would be much lower. In future work, we plan to do scaling experiments and focus on improving precision.

The second task, correctly filtering alarming texts that contain a severe suicide threat, is non-trivial (59.2% F-score). KNN classifiers perform best.

Cascading classifiers is worthwhile, and produces consistently better scores than using one-shot classifiers. It is beneficial for precision in particular, with recall hovering around 50%. Although precision becomes more important as the skewness of the data increases, we would like to improve recall, for example by using only high-recall filters in the first step of the cascade.

Because of the reliance on shallow features, it would be interesting to assess the impact of normalization techniques to correct the often noisy content found in social media. Cleaner input would also allow the use of more advanced features, which require deeper preprocessing.

In conclusion, we believe this work presents a promising approach to suicide prevention in social media, where the potential of using NLP techniques is still largely untapped.

6. Acknowledgements

The work presented in this paper was carried out in the framework of the SubTLe project, funded by the University College Ghent Research Fund. Corpus annotation was done by the Centre for Suicide Prevention (CPZ).

7. References

Christensen, H., Griffiths, K. M., and Jorm, A. F. (2004). Delivering interventions for depression by using the internet: randomised controlled trial. *British Medical Journal*, 328(7434):265, January.

De Smedt, T. and Daelemans, W. (2012). Pattern for Python. *Journal of Machine Learning Research*, 13:2063–2067.

Jashinsky, J., Burton, S. H., Hanson, C. L., West, J., Giraud-Carrier, C., Barnes, M. D., and Argyle, T. (2013). Tracking Suicide Risk Factors Through Twitter in the US. *Crisis*, pages 1–9, October.

Miniño, A. M. and Murphy, S. L. (2012). Death in the United States, 2010. Technical Report 99, National Center for Health Statistics, July.

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

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

Shapero, J. J. (2011). *The Language of Suicide Notes*. Ph.D. thesis.

Stenetorp, P., Pyysalo, S., Topic, G., Ohta, T., Ananiadou, S., and 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*, pages 102–107, Avignon, France.