Journal of the Association for Information Systems (2019) **20**(8)**,** 1075-1128
doi: 10.17705/1jais.00562

RESEARCH PAPER

ISSN 1536-9323

Brute-Force Sentence Pattern Extortion from Harmful Messages for Cyberbullying Detection

Michal Ptaszynski[1], Pawel Lempa[2], Fumito Masui[3], Yasutomo Kimura[4], Rafal Rzepka[5], Kenji Araki[6], Michal Wroczynski[7], Gniewosz Leliwa[8]

[1]Corresponding author, Kitami Institute of Technology, Japan, ptaszynski@cs.kitami-it.ac.jp
[2]Cracow University of Technology, Poland, plempa@pk.edu.pl
[3]Kitami Institute of Technology, Japan, f-masui@cs.kitami-it.ac.jp
[4]Otaru University of Commerce, Japan, kimura@res.otaru-uc.ac.jp
[5]Hokkaido University, Japan, rzepka@ist.hokudai.ac.jp
[6]Hokkaido University, Japan, araki@ist.hokudai.ac.jp
[7]Samurai Labs, Poland, michal.wroczynski@samurailabs.ai
[7]Samurai Labs, Poland, gniewosz.leliwa@samurailabs.ai

Abstract

Cyberbullying, or humiliating people using the Internet, has existed almost since the beginning of Internet communication. The relatively recent introduction of smartphones and tablet computers has caused cyberbullying to evolve into a serious social problem. In Japan, members of a parent-teacher association (PTA) attempted to address the problem by scanning the Internet for cyberbullying entries. To help these PTA members and other interested parties confront this difficult task we propose a novel method for automatic detection of malicious Internet content. This method is based on a combinatorial approach resembling brute-force search algorithms, but applied in language classification. The method extracts sophisticated patterns from sentences and uses them in classification. The experiments performed on actual cyberbullying data reveal an advantage of our method vis-à-vis previous methods. Next, we implemented the method into an application for Android smartphones to automatically detect possible harmful content in messages. The method performed well in the Android environment, but still needs to be optimized for time efficiency in order to be used in practice.

Keywords: Automatic Cyberbullying Detection, Natural Language Processing, Language Combinatorics.

Choon-Ling Sia was the accepting senior editor. This research article was submitted on April 8, 2016 and went through four rounds of revisions.

1 Introduction

Information technology contributions to the preservation, support, and development of public health are numerous. Recent ones along these lines include analysis and prediction of the spread of epidemics (Aramaki, Maskawa, & Morita, 2011), analysis of health data (Buntin, Burke, Hoaglin, & Blumenthal, 2011; Kitajima, Rzepka, & Araki, 2014) and construction of biomedical ontologies (Smith et al. 2005). However, while most of these contributions address the physical sphere of public health, the mental or psychological aspect, although equally important, has been mostly disregarded.

In recent years the problem of unethical behavior in the cyber-environment has been revealed. This has greatly impaired public mental health in adults and especially in children. Specifically, this problem has been termed *cyberbullying*, which is defined as the exploitation of open online means of communication, such as Internet forum boards, or social network services (SNS), in order to convey harmful and disturbing information about private individuals, often children and young adults.

Although attempting to humiliate and slander individuals by means of the Internet has existed almost as long as the Internet itself, with the popularity of devices such as smartphones and tablet computers, cyberbullying can now take place anytime and anywhere. Different access points and the ease of Internet anonymity have further exacerbated this problem.

Messages classifiable as cyberbullying include, for example, ridiculing someone's personality, body, or appearance as well as slandering or spreading rumors. Some cases of cyberbullying have led to victims self-harming or even attempting suicide, or have led to attacks on the offenders. In the US, this issue attracted a great deal of attention beginning in 2006 after a 13-year-old girl from Missouri committed suicide after receiving bullying messages on Myspace.[1] Similar cases have been observed in other countries, including Japan, which is the context of this research. The growing number of cyberbullying cases around the world has stimulated public debate about whether early detection could prevent such tragedies and on the freedom of speech on the Internet, in general (Leets 2001).

In Japan, the problem has become serious enough to garner attention from the Ministry of Education (MEXT 2008). In 2007 Japanese school employees and members of parent-teacher associations (PTA)[2] have started monitoring activities under the general name Internet Patrol ("net-patrol," for short) to detect websites containing such inappropriate content. However, net-patrol is performed manually by volunteers. The vast amount of Internet data make this an uphill battle.

Concern for the victims of cyberbullying motivated us to begin a long-term project that we hope will contribute to detecting, preventing, and ultimately solving the problem of cyberbullying. In the present research we, first, aim to develop a systematic approach to automatically detect and classify cyberbullying entries, which would help and ease the burden of net-patrol members. One of the goals of the project is to create a net-patrol solution by automatically detecting cyberbullying entries on the web and reporting them to the appropriate authorities. Second, we hope to contribute to the prevention of the problem by transferring the cyberbullying detection mechanism onto a mobile device. We developed a test application for Android devices, to test whether it is possible to apply detecting algorithms (typically used on much more powerful machines) on mobile devices such as smartphones. The first results of our study and its possible implications are described in this paper.

The method proposed in this paper is original in the following regards. As previous research has pointed out (Ptaszynski et al. 2010), the language used in cyberbullying messages is often deceptive and messy, and it is difficult to craft a simple set of features to detect it. Therefore, to create a flexible cyberbullying model, we applied a novel automatic feature extraction procedure. In research on machine learning—in particular, research on machine learning used to solve real-world problems—one can use one of two approaches for feature extraction: automatic feature extraction (i.e., *bottom-up approach*) or the selection of custom predefined features (i.e., *top-down approach*) including, for example, creating a lexicon, etc.). The latter approach, although sometimes providing satisfying results, requires deep knowledge of the problem, meaning that the researchers must identify the valid features themselves, which is often inefficient.

In the vast majority of relevant research that applies the bottom-up approach, the features automatically extracted are typically based on separate words (e.g., bag-of-words approaches) (Ptaszynski et al., 2010). Though this strategy employs simple words that could be used to classify text such as parts of speech or concepts (Sahlgren and Cöster 2004), the sophistication of the extracted pattern never exceeds one token. A smaller number of studies applies n-grams (usually unigrams to tetragrams of words or letters) (Damashek 1995, Ponte and Croft 1998, Siu and Ostendorf 2000).

Recently researchers have started to apply skipgrams, which are slightly more generalized versions of n-grams (D. Guthrie, Allison, Liu, L. Guthrie, & Wilks, 2006) that allow one controlled "skip," or a gap. This, however, still falls short of the definition of a pattern for the purpose of our research, in that it allows any number of "skips" with a flexible dynamic distance, which in mathematical terms refers to "ordered combinations without repetitions." Up to this point, this kind of pattern extraction has not been widely applied due to the computational cost it requires. The method proposed here takes advantage of recent computing technologies that offer multiple cores and large amounts of memory to overcome this problem and efficiently compute such patterns. We believe this method of feature extraction will identify features with

[1] https://myspace.com/

[2] An organization composed of parents and school personnel.

a level of sophistication previously unseen in cyberbullying detection research.

In terms of methodology, this paper follows general principles of design science research in information systems (Gregor & Hevner, 2013, Hevner, March, Park, & Ram, 2004) in that it aims to develop a novel artifact (in particular a mechanism for cyberbullying detection in the form of a smartphone application) to address a real-world problem (cyberbullying). The paper also follows the publication schema for a design science research study (Gregor & Hevner, 2013) as outlined below.

First, we define the problem of cyberbullying and present some of the previous related research. We also describe other available cyberbullying detection solutions and explain how our software is different from other cyberbullying detection software available on the market. Next, we describe our method and the data set used in this research. We explain the evaluation settings and thoroughly analyze and discuss the results. Then, we describe the smartphone application we use to implement the mechanism we developed. We describe the functions, elements, and interface of the application. Finally, we describe the preliminary testing intended to verify the performance of the developed application and discuss the test results.

2 Background

2.1 Cyberbullying: A Social Problem

The problem of harmful and offensive messages on the Internet has existed for many years. One reason for such activities is that the anonymity of Internet communication gives users the feeling that malicious behavior will go unpunished. Recently the problem has been officially defined and labeled as cyberbullying. The US National Crime Prevention Council states that cyberbullying happens “when the Internet, cell phones or other devices are used to send or post text or images intended to hurt or embarrass another person” (http://www.ncpc.org/cyberbullying).

Some of the early robust research on cyberbullying was done by Hinduja and Patchin, who performed numerous surveys about the subject in the US (Patchin & Hinduja, 2006, Hinduja & Patchin, 2009). They found that harmful information may include threats, sexual remarks, pejorative labels, and/or false statements aimed at humiliating others. When posted on a social network such as Facebook, Twitter, or an Internet forum, it may disclose humiliating personal data associated with a victim, personally defaming and ridiculing.

From around 2009 to 2011, a number of large-scale questionnaire studies and social campaigns were conducted to measure the occurrence of cyberbullying and to investigate methods of mitigating the problem. For example, Cross et al. (2009) in Australia (Cross et al., 2009) found that cyberbullying affected around 5%-8% of children in Australian schools, depending on the grade. Comparable results have been found in the United States (Kowalski & Limber, 2007), Finland (Sourander et al., 2010), and across Europe (Hasebrink, 2011). Hasebrink et al. (2009), in particular, present even larger estimates, stating that up to one in five young people (not limited to the school environment) have likely experienced bullying or harassment through the Internet or on mobile devices. These estimates have also been confirmed by Li (2007), Pyżalski (2012), and more recently by Kann et al. (2014).

Cyberbullying has also been thoroughly studied and analyzed by Dooley, Pyżalski, and Cross (2009), Dooley et al. (2009), who performed an in-depth comparative analysis of traditional face-to-face bullying and cyberbullying, and by Lazuras, Pyżalski, Barkoukis, and Tsorbazoudis (2012), who discussed the implications of cyberbullying for teachers in school environments. Dooley, Pyżalski, and Cross, in particular, point out some of the similarities between cyberbullying and traditional face-to-face bullying, but also mention some of the differences that make cyberbullying a more difficult problem to contain. The similarities, which contribute to classifying the problem as a type of bullying. For example, both types of bullying involve peer groups—e.g., classmates in face-to-face bullying or “friend groups on social networking sites. Also, all bullying involves repetitive attacks, though cyberbully attacks are often more frequent than face-to-face attacks. Finally, bullying involves an imbalance of power. Typically, one person, or a small group of people, are bullied by a much larger number of bullies. This feature also distinguishes bullying from other types of cyberaggression.

However, the environment and the tools used can also make cyberbullying an even more humiliating experience than its face-to-face counterpart. For example, with the use of Internet, cyberbullying can occur on much larger scale, potentially transforming it into a completely overwhelming experience.

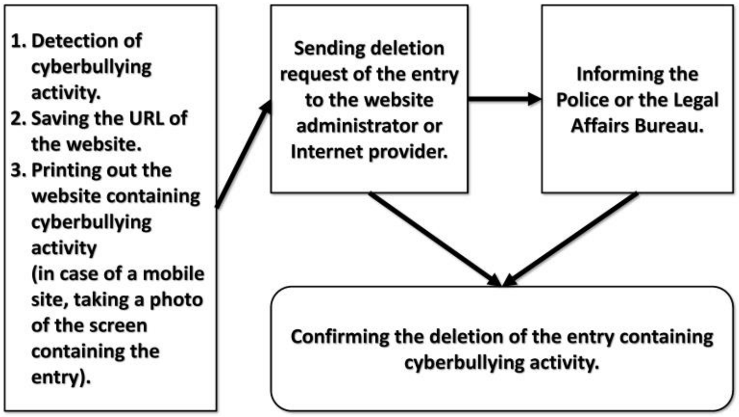

1. Detection of cyberbullying activity.
2. Saving the URL of the website.
3. Printing out the website containing cyberbullying activity (in case of a mobile site, taking a photo of the screen containing the entry).
Sending deletion request of the entry to the website administrator or Internet provider.
Informing the Police or the Legal Affairs Bureau.
Confirming the deletion of the entry containing cyberbullying activity.

Figure 1. Internet Patrol Process

Also, with the popularity of social network services, humiliating a person publicly—for instance, on their official Facebook homepage—may make the attack visible to general public, rather than to a limited number of viewers. In worst-case scenarios, attacks could even appear in search engine results, significantly magnifying its visibility. Additionally, the indirect nature of online relationships is associated with a sense of emotional detachment, making it less likely that Internet bystanders will respond to protect the victim and potentially increasing a bully's sense of impunity.

In Japan, after several cyberbullying victims committed suicide to escape online humiliation, the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) considered the problem serious enough to begin a movement against it. In a manual devoted to handling cyberbullying cases (MEXT, 2008), the ministry places great importance on the early detection of suspicious entries, especially on social networking services and informal school websites. MEXT distinguishes the following several types of cyberbullying detected in Japan.

1. Cyberbullying appearing on BBS forums, blogs and on private profile websites:
   a. Entries containing libelous, slanderous, or abusive contents;
   b. Disclosing personal data of natural persons without their authorization;
   c. Entries and humiliating online activities performed in the name of another person.
2. Cyberbullying appearing in electronic mail:
   a. E-mails directed to a certain person/child, containing libelous, slanderous or abusive contents;
   b. E-mails in the form of chain letters containing libelous, slanderous or abusive contents;
   c. E-mails sent in the name of another person, containing humiliating contents.

For this research we focus mostly on cases of cyberbullying that appear on informal websites associated with Japanese secondary schools. These are websites where pupils exchange information about coursework, tests, etc. However, such pages witnessed a rapid increase in cyberbullying toward pupils and even teachers (Watanabe & Sunayama, 2006), making other users potentially hesitant to use the sites and causing other undesirable consequences.

The Internet Patrol movement was founded to deal with this specific problem. Its participants are typically teachers and PTA members. Based on the MEXT definition of cyberbullying, they read through all Internet contents, and if they find a harmful entry they send a deletion request to the web page administrator and report the event to appropriate authorities, such as the Police or Legal Affairs Bureau.[3] The typical Internet Patrol process is presented in Figure 1.

Unfortunately, net-patrol is presently performed manually as voluntary work. This process includes reading countless Internet entries, determining whether any are potentially harmful, printing out or taking photos of the relevant pages, and sending deletion requests and reports to appropriate organs. With the number of entries growing every day, surveilling the entire web is an uphill battle for the small number of net-patrol members. Moreover, the task potentially places a great mental health burden on the net-patrol members. Our research aims to create a tool allowing for the automatic detection of cyberbullying on the Internet in order to ease the burden carried by net-patrol volunteers and, potentially, other groups with similar goals.

[3] http://www.moj.go.jp/ENGLISH/m_hisho06_00034.html

2.2 Previous Research in Automatic Cyberbullying Detection

Although the problem of cyberbullying has been studied in the social sciences and the field of child psychology for over ten years, there have been only few attempts to detect and study the problem using methods from the field of artificial intelligence (AI) or natural language processing (NLP). Below we present the most relevant research to date and also summarize the publications in Table 1. We mostly focused on journal publications, since they represent the most mature state of research.

The first journal publication written on the topic of automatic cyberbullying detection, cyberbullying analysis, and mitigation using methods from the fields of artificial intelligence, machine learning, and natural language processing was by Ptaszynski et al. (2010). They performed affect analysis of a small data set of cyberbullying entries to find out that a distinctive feature of cyberbullying was the use of profane language . They applied a lexicon of such words to train an SVM classifier. With a number of optimizations, the system was able to detect cyberbullying with 88.2% of the F-score. However, increasing the data diminished their results, resulting in them abandoning SVMs as not ideal for dealing with the frequent language ambiguities that are typical for cyberbullying events.

Later, Ishisaka and Yamamoto (2010) developed a dictionary of abusive expressions based on a large Japanese BBS (electronic bulletin board system, a type of electronic forum) called 2channel. In their research they labeled words and paragraphs that the speaker explicitly used to insult other people—for example, words and phrases like *baka* ("stupid") or *masugomi no kuzu* ("trash of mass media"). Based on which words appeared most often as abusive vocabulary, they extracted abusive expressions from the surrounding context. Unfortunately, their method, based on a 4-gram model did not extract a sufficient number of abusive words, with both Precision and Recall scoring at around 30% or less.

Ikeda, Yanagihara, Matsumoto, and Takishima (2010) manually collected a set of separate harmful and nonharmful sentences. Based on word occurrence within the corpus, they created a list of keywords to classify harmful content. Their method, based on keyword matching in input documents, selected documents as harmful if the number of harmful words found within them was higher than a preset threshold. To deal with the small Recall associated with the method they applied a semantic generalization of documents and based their matching on generalized dependency chunks within input sentences. The highest Precision produced by this method was around 60%, with approximately 35% Recall. Unfortunately, they mostly struggled with variations of the same expressions that differed in only one or two characters—for example, *bakuha* ("blow up") and *baku–ha* ("blooow up"). All variations of the same expression needed to be collected manually, which was a weakness of the method.

Fujii, Ando, and Ito (2010) proposed a system for detecting documents containing excessive sexual language using the concept of distance between two words in a sentence. They defined "black words" as harmful—i.e., words proximal to words that appear only in a harmful context, rather than those that appear in both harmful and nonharmful contexts (i.e., "gray words").

Hashimoto, Kinoshita, and Harada (2010) proposed a method for detecting the harmful meaning of separate words used in jargon. In their method they assumed that the nonstandard meaning of a word is determined by the words surrounding the word in question. They detected the harmful meaning based on calculating the co-occurrence of a word with surrounding words.

In another study, Matsuba, Masui, Kawai, and Isu. (2011) proposed a method to automatically detect harmful entries online that involved extending the SO-PMI-IR score (Turney, 2002) to calculate the relevance of a document with harmful contents. Using a small number of seed words, they were able to apply their method to a large number of documents, predicting which documents were harmful with an accuracy of 83% (based on test data).

Later, Nitta et al. (2013) proposed an improvement to Matsuba et al.'s (2011) method. They grouped the seed words into three categories (abusive, violent, and obscene) to calculate the SO-PMI-IR score and maximized the relevance of the different categories. Their method scored much higher than the original method proposed by Matsuba et al. Nitta et al. based their information retrieval procedure on the Yahoo! search engine API. However, Ptaszynski et al.'s (2016) reevaluation of the method, performed two years after the original paper, showed a major drop in Precision (about 30 percentage points) over two years. Below, in a comparison of our method with previous methods, we discuss possible reasons for such changes, hypothesizing that these changes could have been caused by changes in information available on the Internet (e.g., web page re-rankings, changes in user policies, etc.). In any case, Ptaszynski et al. (2016) achieved considerable success in trying to further improve the method by automatically acquiring and filtering harmful new seed words. Due to similarities in applied data sets and experimental settings, we used all three of the above methods (Matsuba et al. 2011, Nitta et al. 2013, Ptaszynski et al. 2016) as comparisons with the method proposed in this paper.

All the above research was done for Japanese-language applications. For English-language applications, research in the topics related to cyberbullying detection begun around 2012 with Sood, Churchill, and Antin's (2012) research. However, Sood et al. did not yet recognize the full negative weight of harmful online content. The major problem they focused on was personal insults—they assumed that the negative influence of this could, at most, cause the Internet community to stop growing or fall into recession. This would suggest that in 2012, the problem of cyberbullying and its consequences had not yet been fully recognized by the US scientific community. Nevertheless, Sood et al. (2012) focused on the detection of personal insults on the Internet as an aspect of standard community management procedures. Their research used single words and bigrams as features, weighting them using either presence (feature present in input or not 1/0), frequency, or tf-idf, and used them to train an SVM classifier. As a data set they used a corpus of 6,000 entries they had collected from various online forums. As the gold standard for their experiments, they used a crowd-sourcing approach with untrained layperson annotators hired for a classification task through Mechanical Turk.

Later, Dinakar, Jones, Havasi, Lieberman, and Picard (2012) proposed their approach for the detection and mitigation of cyberbullying. An improvement of their study, in comparison to previous research, was its wider perspective. Dinakar et al. (2012) not only focused on the detection of cyberbullying, they also proposed some methods for mitigation. A disadvantage of this research, compared to previous work, was the experimental settings. Dinakar et al. assumed a focus on cyberbullying; however, they did not define the concept strictly enough and, in effect, focused not on the detection of cyberbullying, but rather on detecting entries containing sexual or racial harassment. While these often overlap with cyberbullying, they do not reflect the whole problem. To prepare the data set for experiments, like Sood et al. (2012), Dinker et al. applied Mechanical Turk to entries and comments from YouTube and Formspring, thus formulating the problem as a task for layperson annotators, even though the sophistication of the problem required expert annotators. Despite the lacks concerning overall research settings, the classifiers they used scored up to 58-77% of the F-score depending on the kind of harassment content they detected. Their best proposed classifier was based on support vector machines, which confirmed for English the research done previously by Ptaszynski et al. for Japanese in 2010.

At the same time that Nitta et al. (2013) proposed their extended SO-PMI-IR method for cyberbullying detection, Cano, He, Liu, and Zhao (2013) proposed their violence detection model, a weakly supervised Bayesian model. They did not, however, focus strictly on cyberbullying, but widened their scope to more generally understood "violence." This approach to problem formulation made it understandable, thus making it feasible for annotation by laypersons, allowing them to study the problem without needing to consult experts for help annotating their data sets. The training data sets were extracted from violence-related topics on Twitter and DBPedia and the model was tested on Twitter.

Marathe & Shirsat (2015) applied a Naive Bayes classifier to detect cyberbullying comments on YouTube. They first searched for videos promoting cyberbullying (the search was done subjectively). Next, they extracted features related to those videos, such as video metadata (time stamp, duration, popularity), bag-of-words extracted from video title, description, comments, and profiles of users who uploaded the videos. Then, they built a character n-gram model based on such features and trained the Naive Bayes classifier to detect whether a new input video (or its metadata) relates to cyberbullying. Although neither their data collection approach nor their results could be considered state of the art, the idea to include a context wider than a simple bag-of-words approach is noteworthy.

Sarna & Bhatia (2017) proposed categorization of cyberbullying messages into direct and indirect bullying. They based their method on a set of features like "bad words," as well as words indicating positive or negative sentiment and other common features like pronouns and proper nouns to estimate user credibility. They used a top-down approach for feature extraction and assumed some features correlated with cyberbullying (e.g., "bad words" or emotion words). They applied those features to classify messages into direct bullying, indirect bullying, and nonbullying with the use of four standard classifiers (Naive Bayes, kNN, decision trees, SVM). The results of the classification were further used in the user behavior analysis model, which provided the output for analysis of user credibility. An interesting part of their research was that they immediately recognized that cyberbullying can be performed directly in the form of insults as well as indirectly, in other linguistic forms like irony, jokes, or rumors. Unfortunately, in practice, Sarna & Bhatia (2017) only focused on messages that contained "bad words" (which they also failed to specify in the paper), thus undermining their primary assumption. The problem with their research was that they did not provide any detailed information about the data set they used (except that it was from Twitter), nor did they explain how they performed the annotation of the data set (by expert or layperson annotators). Therefore, a direct comparison to their method, as well as its objective evaluation, remains problematic.

Table 1. Summary of Previous Research in Cyberbullying Detection.

References	Processing language	Feature extraction method	Classification method
Ptaszynski et al., 2010	Japanese	Unigrams (BoW)	SVM
Ishisaka & Yamamoto, 2010	Japanese	4-grams	n-gram language model matching
Ikeda et al., 2010	Japanese	Unigrams (harmful single words)	keyword matching
Fujii et al., 2010	Japanese	Unigrams co-occurring with sexting "black" and "gray" words	keyword matching
Hashimoto et al., 2010	Japanese	Separate words	surrounding word co-occurrence with harmful words
Matsuba et al., 2011	Japanese	9-seed words	SO-PMI-IR averaged for all seed words
Sood et al., 2012	English	Unigrams, bigrams, stems	SVM, various weighting (presence, freq, tf-idf)
Dinakar et al., 2012	English	Unigrams, handcrafted word lists (Ortony lexicon of negative words, profane words, frequently occurring stereotypical words), POS	SVM, JRip, Naive Bayes, J48
Nitta et al., 2013	Japanese	seed words grouped into three categories	SO-PMI-IR maximized for category
Cano et al., 2013	English	Violence-related words (derived from violence-related topics from Twitter and DBPedia)	violence detection model (weakly supervised Bayesian model)
Marathe & Shirsat, 2015	English	BoW, video metadata (time stamp, popularity), character n-grams	Naive Bayes
Ptaszynski et al., 2016	Japanese	Seed words grouped into three categories	SO-PMI-IR maximized for category with seed word optimization
Sarna & Bhatia, 2017	English (?)	"Bad words," positive- and negative-sentiment words, pronouns, proper nouns, links	Naive Bayes, kNN, decision trees, SVM

2.2.1 Research Gaps

Based on the above literary review, we recognized the following research gaps, or areas for improvement, in cyberbullying detection research:

(1) Data set preparation. Most of the abovementioned methods suffer from unprofessional or subjective data preparation. In some research, such as that by Cano et al. (2013) or Dinakar et al. (2012), the data sets were indeed collected with sufficient scrutiny (e.g., using Mechanical Turk or applying an already existing available data set), although they did not define the problem ideally. The problem of cyberbullying is a complex social phenomenon and data representing its samples need to be handled by expert annotators. Some researchers, like Ishisaka & Yamamoto (2010), Sood et al. (2012) or Cano et al. (2013) recognize this difficulty in defining the problem, and formulate it in other terms, so the use of experts is not needed, and thus focus on, for example, detecting generally perceived violence or aggression. Some researchers use the term cyberbullying, although what they actually focus on—for example, sexual or racial harassment—overlaps with but is not equivalent to cyberbullying. On the other hand, other research, such as Sarna & Bhatia (2017) or Marathe & Shirsat (2015), collects the data sets ad hoc and with no specific standardization. Of all the relevant research, the studies by Ptaszynski et al. (2010), Matsuba et al. (2011), Nitta et al. (2013) and Ptaszynski et al. (2016) were the only ones that defined the cyberbullying problem with proper depth and made the effort to collect data sets using sufficient scrutiny and standardization. They obtained their data sets from an official source—namely, from a Japanese branch of the Human Rights Center, which collected the data fully labeled by Internet Patrol members actively involved in searching for cyberbullying cases on the Internet, based on an official governmental definition of cyberbullying (MEXT 2008).

(2) Feature extraction. Almost all of the reviewed research included only words (tokens, unigrams), or n-grams as features, at best. Some research (Matsuba et al.2011, Nitta et al. 2013) applied only a small number of features, while others (Dinakar et al. 2012, Marathe & Shirsat 2015) built up more complex models, which, however, still did not exceed a simple bag-of-words model. Moreover, research, such as Matsuba et al. (2011), Ishisaka & Yamamoto (2010), Nitta et al. (2013), Sarna & Bhatia (2017) used only top-down selected features. While, to some extent, it is reasonable to apply top-down features in automatic cyberbullying detection—for example, in the case of violent or obscene words—top-down feature selection requires time and extensive effort and background knowledge about the data set, and thus tends to be inefficient with limited practicality. Moreover, some research disregarded significant findings of other previous research. For example, Ptaszynski et al. (2010) recognized with sufficient certainty that emotion-related words are not effective in cyberbullying detection, but Sarna & Bhatia (2017) still put great weight on using positive and negative sentiment words in their classification. To allow full comparison to previous research, our research also allows for simple bag-of-words and n-gram models. However, in contrast to all previous research, we also apply a novel and sophisticated idea of combinatorial patterns extracted automatically from training sentences.

(3) Classification methods. Previous research tested many different classifiers, such as SVMs, Naive Bayes, or decision trees. However, in most cases, SVMs attained the highest scores. Interestingly, despite Ptaszynski et al. (2010) concluding that SVMs, especially those trained on BoW, are not ideal for grasping the sophistication of language used in cyberbullying, in most of later research, when comparing various classifiers, the researchers obtained their highest scores precisely for SVMs. Therefore, this research also used SVMs for comparison with our proposed method. Furthermore, one major disadvantage of traditional machine learning is the fact that although one can set groups of specific features that influence the results, it is not possible to specify which feature in which case scenario influenced the acquired result. The method proposed in this paper is capable of providing such information, which could be useful in further filtering and optimizing the features for other specific cyberbullying data sets.

(4) Human effort reduction. In our research we aimed at minimizing human effort. Most of the previous studies assumed that using vulgar words as seeds would help detect cyberbullying, even though they all recognized that vulgar words are only one kind of distinctive vocabulary and do not cover all cases. We assumed that this kind of vocabulary could be extracted automatically. Moreover, we did not restrict the scope to words (unigrams, tokens), or even phrases (n-grams). We extended the search to sophisticated patterns with disjointed elements. To achieve this, we developed a pattern extraction method based on the idea of a brute-force search algorithm, which, although being computationally heavy in the training phase, removes most of the workload from the researchers and provides a method capable of replacing the efforts of Internet Patrol members seeking Internet cyberbullying entries.

2.3 Related Work in the Development of Practical Applications

With the popularization of mobile devices, the problem of cyberbullying has become increasingly apparent. Apart from the research in automatic cyberbullying detection described above, a number of research teams around the world have attempted to develop practical solutions for the detection and mitigation of this problem (Ptaszynski et al., 2010; Dinakar et al., 2012; Nitta et al., 2013; Kontostathis, Reynolds, Garron, and Edwards, 2013). However, most of the research is still in a developmental phase and is yet to be fully applied in practice. On the other hand, market software solutions for the detection and mitigation of online bullying have been developed and have the potential capacity to deal with the problem to some extent. Unfortunately, such solutions are usually based on simple methods, thus narrowing the scope of their applicability. Below we summarize some of these:

FearNot! One example of a software using a novel approach is FearNot![4] The authors describe it as "an interactive drama/video game that teaches children strategies to prevent bullying and social exclusion." The development of this software, which uses a psychology-inspired AI character, was supported by the EU-funded research projects Victec and eCircus. The approach taken by the developers—namely, to not detect and stigmatize cyberbullying behavior but, rather, to educate children on how not to become bullies, does indicate deep insight into the problem. Unfortunately, the development of the software stopped in early 2013.

BullyGuardPro. BullyGuardPro[5] is an example of a potentially effective software solution that is aimed at detecting cyberbullying activity around a user by allowing the user to "effectively respond, diffuse and halt cyberbullying and cyberpredation attacks." The software was developed by Lynne Edwards and April Kontostathis, who lead one of the first teams to research cyberbullying detection (Kontostathis et al., 2013). Unfortunately, at the time of writing, no details on the technology used in the software or its release date were available.

Samaritans Radar. On October 29, 2014, "Samaritans,"[6] an organization focused on suicide prevention, launched an application called Samaritans Radar. It was a free Internet application for Twitter, which helped users monitor their friends' tweets. The

[4] https://sourceforge.net/projects/fearnot/
[5] http://www.bullyguardpro.com
[6] http://www.samaritans.org/

main function of the application was to alert users if it detected anyone in a user's online surroundings, who might be a bullying victim, depressed, or who was sending disturbing suicidal signals. Unfortunately, due to serious data protection and privacy issues, recognized by users soon after the launch, the application closed permanently on March 2015.

Uonevu. An interesting approach to the detection of cyberbullying in messages has been developed by researchers from Trinity College, Dublin and the National Anti-Bullying Research and Support Centre, under the codename Uonevu (meaning "bullying" in Swahili). The software is meant to detect particularly nonliteral forms of bullying and negative stereotyping. The software works by applying a semantic knowledge base in order to associate concepts with each other. An example is associating the concept of "fat/obese people" with the word "pizza," which in a sentence such as "Hey, Jane, are you going to eat a whole pizza tonight?" would indicate cyberbullying. The major problem here becomes creating a large enough knowledge base of stereotypes. Unfortunately, at the time of writing, the database contained only 57 stereotypes, and is thus not sufficient for the effective functioning of the software. However, the project is still in its developmental phase and could be an interesting solution once finished.

Twitter New Policy. On February 26, 2015, Twitter independently released its new policy regarding safety and misbehavior among its users.[7] Twitter allowed users to report particular tweets as harassment incidents. This provided users who became victims of online bullying with a tool to personally respond to bullying attacks. The account of a confirmed bully is locked and can be reopened only under the condition that the bully deletes any harmful tweets. This way of dealing with cyberbullying is not yet a software per se, but Twitter aims to detect bullying messages automatically in the future. For the time being, however, they are increasing the number of support staff devoted to handling abuse reports.

ReThink. An example of a recent popular solution that would still work today is ReThink,[8] an application for smartphones that shows a pop-up warning message when user tries to send a message containing harmful content. The idea of informing a user about the possible harmfulness of a message has been recognized in the research as an effective means of making a user reevaluate his or her message before making it publicly available (Masui et al., 2013; Patent Application No. 2013-245813). Unfortunately, although ReThink is a good example of a quick and ad hoc response to the cyberbullying problem—its algorithm for the detection of harmful contents is based on simple keystroke logging and detecting vulgar and harmful words within a string of characters. This makes it incapable of detecting more sophisticated contents that do not include vulgar expressions. It also fails when user makes a mistake during writing and, for example, uses a backspace, since the "backspace" character is also recorded and hinders the detection of harmful words.

PocketGuardian. PocketGuardian is a newer (released September 2015) example of software for parental monitoring, which "detects cyberbullying, sexting, and explicit images on children's mobile devices."[9] By using machine learning techniques, the software provides a statistical probability that the content (sentence, tweet, e-mail, or image) is inappropriate. An advantage of this software is that it focuses not only on textual content but also includes in its monitoring range the ability to detect explicit images. A disadvantage could be its price ($12.99 per month). The exact technology behind the software (e.g., applied machine learning algorithms, size of the training lexicon or corpus) is yet unknown. Moreover, as the software is aimed at parents trying to monitor their children's mobile devices, the developers will need to address questions regarding ethics of using such software and its influence on the parent-child trust relationship.

In comparison with the software described above, the application presented later in this paper distinguishes itself in the following ways. Similarly to ReThink, it provides a tool for the user to reflect on their own written messages. However, in contrast to ReThink, it shows the user which exact words or sentence patterns were considered inappropriate. Moreover, it uses not only simple keystroke logs, but also various artificial intelligence methods (at present, two) to spot any undesirable contents. Our application focuses only on textual contents—however, in contrast to PocketGuardian, we do not intend to make a profit off the application. Moreover, since the application is intended to be employed by the user directly, all ethical issues and any influence on parent-child trust relations—as well as any privacy issues, like those confronted in the Samaritans Radar application—are unlikely to arise. The method we propose has been under development for over six years—thus, the problem of insufficient data, plaguing, for example, the Uonevu project, is also resolved.

[7] https://blog.twitter.com/2015/update-on-user-safety-features

[8] http://www.rethinkwords.com/

[9] https://gopocketguardian.com/

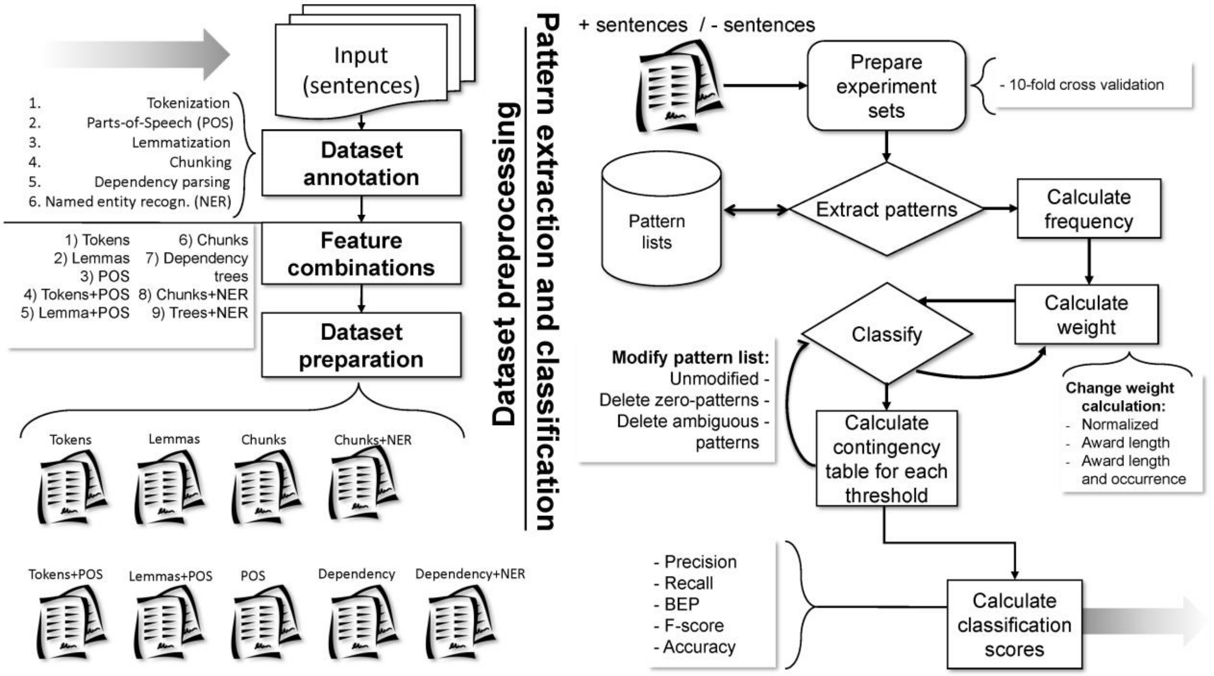


Figure 2. A Graphical Summary of the Whole Method.

3 Sentence Pattern Extortion Method Description

In this section we explain all parts of the proposed method, step by step. At first, we describe general types of features we apply in this research and explain our feature extraction method. Next, we describe all methods of weight calculation for the applied features. Then, we present the applied classifiers. Finally, we describe the threshold optimization as a method to optimize the classifier performance. A graphical summary of the whole method is presented in Figure 2.

3.1 Sophistication of Language Model

Features applied in building a language model for classification can, in general, be viewed from two different points of view. First, the general sophistication of all features must be addressed. This applies to whether the applied features represent single words/tokens, or n-grams (sequences of tokens), or some other more sophisticated kinds of patterns. Second, a specific kind of information encoded in features needs to be recognized. For example, features can consist of words, but also of lemmas (undeclined dictionary form of words), parts of speech (POS), etc. Combinations of those types of information are also allowed. For example, it is possible to use combined features of words with POS, or lemmas with POS. Using lemmas with POS, being more related to the data set rather than to the method itself, will be explained further in Section 3.2 below. We first turn our attention to using combined features of words with POS, as this constitutes the core of the method.

The computationally simplest language model is called bag-of-words (BoW) (Harris, 1954). It considers a piece of text or document as an unordered collection of words, thus disregarding grammar and word order. Although recently a generalization of the BoW model has been proposed using semantic concepts instead of words (bag-of-concepts) (Cambria & Hussain 2012, Raymond et al. 2012) or an aspect-query model (Song, Huang, Bruza, & Lau, 2012), the general rule remains the same—namely, that the order of elements within the input, as well as longer strings of elements (e.g., phrases), are disregarded.

One important approach retaining the significance of word order is broadly called the n-gram approach. In terms of probabilistic theory, its basis was first formulated by Markov (1971). The n-gram approach perceives a given input (e.g., a sentence) as a set of n-long ordered subsequences of words. This allows matching the words while retaining the sentence word order. However, the n-gram approach, when applied to language, still allows only for simple sequence matching, disregarding more sophisticated sentence structure.

Table 2. Comparison of Capabilities of Different Language Models oo Capture Certain Patterns from a Hypothetical Corpus Consisting of Two Sentences, (1) and (2)[a]

Pattern	Model			
	BoW	N-gram	Skip-gram	LC
John	○	○	○	○
John went	×	○	○	○
John * to	×	×	○	○
John * school	×	×	×	○
John * to * today	×	×	×	○

Note: ○ = capable, × = incapable.
[a]Sentence (1) and (2) are as follows:
(1) *John went to school today.*
(2) *John went to this awful place many people tend to generously call school today.*

An example of such a sophisticated pattern can be explained as follows. The following sentence (in Japanese) *Kyō wa nante kimochi ii hi nanda !* (What a pleasant day it is today!) contains a common and widely studied language pattern *nante * nanda !*[10] Similar cases can be easily found in other languages; for instance, in English, the exclamative sentence "Oh, she is so pretty, isn't she?" contains a pattern "Oh * is so * isn't *?," which is a typical example of a *wh*-exclamative sentence pattern (Beijer, 2002, Potts & Schwarz, 2008). The existence of such patterns in language is common and well recognized. However, it is not possible to discover them using an n-gram approach.

An example of a language model that aims to go beyond BoW and n-grams is the skip-gram model (sometimes also called skipped n-gram or distanced n-gram). It assumes that some words within an n-gram might not be adjacent, that they are skipped over. In theory, this should allow extraction of most of frequent language patterns from a corpus. However, there are some major drawbacks in research studying skip-gram modeling. These include, for example, assuming that a skip can appear only in one place (Huang, Alleva, Hon, Hwang, & Rosenfeld, 1992). The above-mentioned English sentence example clearly indicates that frequent and easily recognizable language patterns can consist of elements appearing variously at the beginning of a sentence, in the middle of a sentence, or at the end of the sentence, depending on the situation. Multiple gaps between them are also common, and these are not covered by the skip-gram language model. Moreover, the number of skipped elements is recorded for each gap. For example, a 2-skip-3-gram can only allow 2 skips (omitting two words) between the elements, which means that the model considers as different patterns two cases in which the first gap has 2 skips and the second has 5 skips. Since the model assumes full control of the skip-length, the 2-skip-3-gram and 5-skip-3-gram consisting of the same elements (words) are represented as different entities and can never refer to the same pattern in a corpus. This assumption is unrealistic, since one can easily imagine that the same pattern, appearing in two sentences of different lengths, will be separated by gaps of different sizes. To illustrate this problem, in Table 2 we compare which of the above-mentioned language models are capable of discovering particular patterns present in the two sentences below. The last column on the right represents the capability of the language model based on the idea of language combinatorics (LC), applied in this research.

(1) *John went to school today.*

(2) *John went to this awful place many people tend to generously call school today.*

The language modeling method discussed in this paper is capable of dealing with any of the sophisticated patterns. This is due to the fact that we define sentence pattern as any ordered nonrepeated frequently occurring combination of sentence elements. This definition allows extraction of all possible frequent[11] meaningful linguistic patterns from unrestricted text.

[10] Equivalent to *wh*-exclamatives in English (Sasai 2006, Beijer 2002); asterisk "*" used as a marker of disjointed elements.

[11] In this research "frequent pattern" means a combination occurring in a corpus at least twice. This differs from the traditional approach to building BoW language models, where all extracted words are generally used, even if their occurrence is equal to 1 because single words usually do not have high occurrence rates and most of such cases are rare. Therefore, not using all words, or using only those with occurrence = 2 or higher, could significantly decrease the Recall rate in the classification process (not many features would be found). With sophisticated patterns, however, words are extracted in large numbers and using all of them would make the classification process inefficient. By using the above cut-off rate, we conform to the general definition of "a pattern" (as something that appears "at least twice"), eliminate the least useful patterns, and retain those which appear most often.

3.2 Feature Extraction with Language Combinatorics

To extract the patterns of cyberbullying messages we first applied the idea of *language combinatorics* (Ptaszynski et al. 2011). This idea assumes that linguistic entities such as sentences can be perceived as bundles of ordered nonrepeated combinations of elements (words, punctuation marks, etc.). Furthermore, the most frequent combinations appearing in many different sentences can be defined as sentence patterns.

We assumed that for the task of cyberbullying detection, where actual harmful meaning is often hidden and indirect, applying sophisticated patterns with disjointed elements should provide better results than the usual bag-of-words or n-gram approach. As long as patterns are defined as ordered combinations of sentence elements, they could be automatically extracted by generating all ordered combinations of sentence elements, verifying their occurrences within a corpus, and filtering out those combinations which appear only once.

Algorithms using such combinatorial approaches initially generate a massive number of combinations / potential answers to a given problem. This is the reason they are sometimes called brute-force search algorithms. The brute-force approach often faces the problem of exponential and rapid growth of function values during combinatorial manipulations. This phenomenon is known as combinatorial explosion (Krippendorff, 1986). Since this phenomenon often results in very long processing time, combinatorial approaches have been often disregarded. We assumed however, that combinatorial explosion can be dealt with on modern hardware to the extent needed in our research. Moreover, optimizing the combinatorial approach algorithm specifically to the problem requirements should shorten the processing time making it advantageous in the task of processing harmful language.

From the fact that the method first extracts all possible patterns from a sentence with a brute-force-inspired algorithm, we call the method ***pattern extortion***, to distinguish it from typical *pattern extraction* methods based on n-grams or single tokens.

In particular, this method, first, orders nonrepeated combinations generated from all elements of all input sentences in a training set. In every n-element sentence there is k-number of combination clusters, such as that $1 \leqq k \leqq n$, where k represents all k-element combinations being a subset of n. The number of combinations generated for one k-element cluster of combinations is equal to a binomial coefficient, represented in Equation 1. In this procedure, the system creates all combinations for all values of k from the range of $\{1, \dots, n\}$. Therefore, the number of all combinations is equal to the sum of combinations from all k-element clusters of combinations, like in Equation 2.

(1)

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$$

(2)

$$\sum_{k=1}^{n} \binom{n}{k} = \frac{n!}{1!\,(n-1)!} + \frac{n!}{2!\,(n-2)!} + \cdots + \frac{n!}{n!\,(n-n)!} = 2^n - 1$$

Next, all nonsubsequent elements are separated with an asterisk (“*”). All patterns generated this way are used to extract frequent patterns appearing in a given corpus. Exact examples of patterns created from one sentence are represented in Figure 3.

For comparison, we also applied more traditional n-gram and BoW-based language models. In a classification experiment we used BoW for traditional classifiers applied in previous research, and n-grams and sophisticated patterns for the proposed classifier. It was not possible to apply sophisticated patterns in traditional classifiers such as SVMs because of an incomparably large number of patterns generated by the proposed methods, as compared to only single words in BoW model. Consequently, great amounts of computational power would be required, making the method unpractical and inefficient, especially since the exact patterns influencing the results would still not be accessible with SVMs and other classifiers.

Example: What a nice day !

5-el. pattern:	**4-el. pattern:**	**3-el. pattern:**	**2-el. pattern:**	**1-el. pattern:**
What a nice day !	What a nice * !	a nice * !	What a	What
	What a nice day *	What a nice	What * !	a
	What a * day !	What a * !	nice * !	nice
	…	…	…	…
no. of patterns: (1)	(5)	(10)	(10)	(5)

Figure 3. Examples of Various Length (i.e., Number of Elements) Patterns Extracted from One Sentence.

3.3 Weight Calculation

After combinatorial patterns are extracted, their occurrences O are calculated separately for the positive side (positive, meaning "harmful") O_{pos} and the negative side (negative, meaning "nonharmful") side O_{neg}. The occurrences of each pattern j are further used to calculate normalized pattern weight w_j according to Equation 3, which is a simplified *sigmoid* function normalizing the weight score between 1 (completely harmful) and -1 (completely nonharmful).

$$w_j = \left(\frac{O_{pos}}{O_{pos} + O_{neg}} - 0.5\right) * 2 \tag{3}$$

The weight can be calculated in several ways. Two features are important in weight calculation. A pattern is more representative for a corpus when it is, first, longer (length k) and, second, appears frequently in the corpus (occurrence O). Thus, the weight can be modified by

- awarding length (LA) by multiplying normalized weight w_j by pattern length k_j , which provides a weight with awarded length w_{LA}, like in Equation 4, or
- awarding length and occurrence (LOA) by multiplying normalized weight w_j by pattern length k_j and overall pattern occurrence (O_{pos} +O_{neg}), which provides a weight with awarded length and occurrence w_{LOA}, like in Equation 5.

$$w_{LA} = w_j * k_j \tag{4}$$

$$w_{LOA} = w_j * k_j * (O_{pos} + O_{neg}) \tag{5}$$

The list of frequent patterns generated in the process of pattern generation and extraction can be also further modified. When two collections of sentences of opposite features (such as "positive vs. negative" or "harmful vs. nonharmful") are compared, a generated list of patterns will contain patterns that appear uniquely on only one side (e.g., uniquely positive patterns and uniquely negative patterns) or on both sides (i.e., ambiguous patterns). Therefore, the pattern list can be further modified by

- erasing all ambiguous patterns[12] (AMB), or
- erasing only ambiguous patterns which appear in the same number on both sides ("zero-patterns," or 0P).

Moreover, a list of patterns will contain both the sophisticated patterns (with disjointed elements) as well as more common n-grams. Therefore, the experiments on the proposed method were performed with both

- patterns (PAT), and
- n-grams (NGR) only.[13]

For the model based on BoW, we also applied a traditional weight calculation scheme—namely, term frequency divided by document frequency (tf*idf). Term frequency *tf(t,d)* refers here to the traditional raw frequency, meaning the number of times a term t (word, token) occurs in a document d. Inverse document frequency *idf(t,D)* is the logarithm of the total number of documents |D| in the corpus divided by the number of documents containing the term n_t , as in Equation 6.

$$idf(t,D) = log\frac{|D|}{n_t} \tag{6}$$

Finally, *tf*idf* refers to term frequency divided by document frequency or, in other words, multiplied by inverse document frequency.

3.4 Classification

The proposed classifier is a function defined as a sum of weights of patterns found in the sentence, like in Equation 7.

$$\text{score} = \sum w_j\,, (1 \geq w_j \geq -1) \tag{7}$$

It produces a harmfulness score for each analyzed sentence. The score alone does not yet specify whether a sentence can be considered harmful or not; however,

[12] By ambiguous patterns we mean those patterns that appear on both the harmful side and the nonharmful side. If a pattern appears on both sides it is ambiguous (more ambiguous, or less ambiguous depending on the score) whether it is harmful or nonharmful and its normalized weight is in the range of 0.99(9) to -0.99(9). Moreover, if an ambiguous pattern appears on both sides in the same occurrence, then its weight is equal 0, and is thus called a zero-pattern.

[13] We also performed the classification with the use of skip-grams. Unfortunately, skip-grams failed completely. Due to their construction, according to which even almost identical patterns are considered as completely different if the number of skips is different, or the skip-length is different, skip-grams need a very large data set to collect enough even somewhat frequent features (actual skip-gram instances). With our assumption that a pattern is "something that appears at least twice in a corpus," there were no skip-grams available to be used in classification.

a good guess is that the more *above* zero the score, the more harmful patterns it contains or, to put it more clearly, the more it resembles a style of writing usually found in cyberbullying. On the other hand, the more *below* zero the score, the more it resembles a nonharmful way of writing. However, this instinctive rule of thumb, with zero as a universal threshold, does not apply to pattern-based methods, since even a one-word difference in a sentence can produce a much larger number of patterns on one of the sides (harmful or nonharmful), thus causing an imbalance in the data. Therefore, we also performed a threshold optimization to specify which threshold should be used for the applied data.

Apart from the above, for the classification we also used other methods based on various classifiers for comparison, such as SVM, Naive Bayes, JRip, J48, or kNN, applied previously in other research (Ptaszynski et al., 2010; Sood et al., 2012; Dinakar et al., 2012; Sarna & Bhatia, 2017; Marathe & Shirsat, 2015), and SO-PMI-IR, which has also frequently been applied in previous research (Matsuba et al., 2011; Nitta et al., 2013; Ptaszynski et al., 2016).

3.5 Threshold Optimization and Heuristic Rules

If the initial collection of sentences is biased toward one of the sides (e.g., more sentences of one kind, or the sentences were longer, etc.), there will be significantly more patterns of a certain kind. Thus, to avoid bias in the results, instead of applying a static rule of thumb, the threshold was optimized automatically.

All the above settings were automatically verified in the process of evaluation, based on 10-fold cross-validation, in order to choose the best model. The training in 10-fold cross-validation was done separately for each fold. Namely, the training was performed ten times, each time on different data parts and with test data completely unrelated to the training data. The metrics used in evaluation were standard: Precision (P), Recall (R), balanced F-score (F), and Accuracy (A). These scores were calculated for every threshold and compared to choose the optimal model.

Finally, to deal with the combinatorial explosion mentioned at the beginning of this section we applied two heuristic rules. In the preliminary experiments we found that the most valuable (frequently appearing) patterns in language are up to six elements long. Thus, we limited the scope of pattern extraction to $k \leq 6$. As such, the procedure of pattern generation (1) generates up to six element-patterns, or (2) terminates at the point where no frequent patterns were found.

4 Evaluation Experiment

4.1 Data Set

We first needed to prepare a data set. We used the data set created originally by Matsuba, Masui, Kawai, Isu (2010) and developed further by Matsuba et al. (2011). The data set was also used by Ptaszynski et al. (2010), Nitta et al. (2013), and recently by Ptaszynski et al. (2016). It contains 1,490 harmful and 1,508 nonharmful entries. The original data were provided by the Human Rights Research Institute Against All Forms for Discrimination and Racism in Mie Prefecture, Japan[14] and contains data from unofficial school websites and forums. The harmful and nonharmful sentences were manually labeled by Internet Patrol members according to official instructions included in the MEXT manual for dealing with cyberbullying (MEXT 2008). Some of these instructions are briefly summarized below.

The MEXT definition assumes *that cyberbullying happens when a person is personally offended online*. This may include disclosing the person's name, personal information, and other details related to privacy concerns. Therefore, as the first feature distinguishable for cyberbullying, MEXT defines private names. This includes information such as:

- Private names and surnames (e.g. "Michal Ptaszynski"),
    - ◦ When a person's name can be clearly distinguished
- Initials and nicknames (e.g. "Mr. P.", "Mi*al Ptasz*ski")
    - ◦ When a person's identity can be clearly distinguished
    - ◦ When a person's identity cannot be clearly distinguished
- Names of institutions and affiliations (e.g. "That foreigner professor from Kitami Institute of Technology")
    - ◦ When a person's identity can be clearly distinguished
    - ◦ When a person's identity cannot be clearly distinguished

As the second feature distinguishable for cyberbullying, MEXT defines any other type of personal information. This includes:

- Address, phone numbers, etc. (e.g. "165 Koen-cho, Kitami, 090-8507, Japan", or "+81-157-26-9327")

[14] http://www.pref.mie.lg.jp/jinkenc/hp/

    - When the information refers to a private person
    - When the information is public or refers to a public entity
- Questions about private persons (e.g. “Who is that tall foreigner wandering around in the Computer Science Dept. hallways?”)
    - Always considered undesirable and harmful, even if the object is described in a positive way, because such questions can lead to rumors.
- Entries revealing personal information (e.g. “I heard that guy is responsible for the new project.”).
    - When a person’s identity can be clearly distinguished
    - When a person’s identity cannot be clearly distinguished

Furthermore, literature on cyberbullying indicates vulgarities as one of the most distinctive features of cyberbullying (Patchin & Hinduja 2006, Hinduja & Patchin 2009, Ptaszynski et al. 2010). Also, according to MEXT, vulgar language is often used in the context of cyberbullying due to its ability to convey offenses against particular individuals. Examples of such words in English are *shit*, *fuck*, and *bitch*. Examples in Japanese are *uzai* (freaking annoying) and *kimoi* (freaking ugly). In the prepared data set all entries containing any of the above information was classified as harmful. Some examples from the data set are represented in Table 3.

4.2 Data Set Preprocessing: Feature Selection

The proposed method uses sentences separated into elements (words, tokens, etc.) as an input. In the transcription of Japanese (the applied data set language) spaces (e.g., between words) are not used as they would be in English, for example. Therefore, we needed to preprocess the data set and make the sentences separable into elements. We did this in several ways to determine how the preprocessing would influence the results. We used MeCab,[15] a morphological analyzer for Japanese and CaboCha,[16] a Japanese dependency structure analyzer, to preprocess the sentences from the data set in the following ways.

- **Tokenization**: All words, punctuation marks, etc. are separated by spaces (TOK).
- **Lemmatization**: As above, but words are represented in their generic (dictionary) forms, or “lemmas” (LEM).
- **Parts of speech**: Words are replaced with their representative parts of speech (POS).
- **Tokens with POS**: Both words and POS information are included in one element (TOK+POS).
- **Lemmas with POS**: As above but with lemmas instead of words (LEM+POS).
- **Chunks**: Larger subparts of sentences divided by grammatical cluster—e.g., noun phrase, verb phrase, predicate, etc.—but without dependency relations (CHUNK).
- **Dependency structure**: As above, but with information regarding how a chunk relates to the previous chunk, the following chunk, and to other chunks (DEP).
- **Chunks with named entities**: Chunks with added information on what *named entities* (private name of a person, organization, numbers, etc.) appear in the sentence. The information is provided by the dependency structure analyzer (CHUNK+NER).
- **Dependency structure with named entities**: Both dependency relations and named entities are provided (DEP+NER).

Feature extraction from sentences is performed automatically, according to procedures explained in Section 3.2. Next, the language model is generated automatically using the extracted features. In this context, the data set preprocessing methods represented above can be understood as a feature selection preset for the experiment.

Five examples of preprocessing are represented in Table 4. Theoretically, the more generalized a sentence is, the less unique patterns it will produce, but the produced patterns will be more frequent. This can be explained by comparing a tokenized sentence with its POS representation. For example, in the sentence from Table 4 we can see that a simple phrase *kimochi_ii hi* (“pleasant day”) is represented by a POS pattern as ADJ N. We can easily assume that there will be more ADJ N patterns than *kimochi_ii hi*, because many word combinations can be represented by this POS pattern. On the other hand, there are more words in the dictionary than POS labels. Therefore, POS patterns will be less varied but will occur more frequently. By comparing the results of the classification using different preprocessing methods, we can find out whether it is better to represent sentences as more generalized or as more specific.

[15] http://taku910.github.io/mecab/

[16] https://taku910.github.io/cabocha/

Table 3. Four Examples of Cyberbullying Entries Gathered During Internet Patrol.

>>104 Senzuri koi te shinu nante? sonna hageshii senzuri sugee naa. “Senzuri masutaa” toshite isshou agamete yaru yo. >>104 Dying by “flicking the bean”? Can’t imagine how one could do it so fiercely. I’m gonna worship her as a “master-bator,” that’s for sure.
2-nen no tsutsuji no onna meccha busu suki na hito barashimashoka? 1-nen no anoko desuyo ne? kimogatterunde yamete agete kudasai Wanna know who likes that awfully ugly 2nd-grade Azalea girl? It’s that 1st-grader isn’t it? He’s disgusting, so let’s leave him mercifully in peace.
Aitsu wa busakute sega takai dake no onna, busakute se takai dake ya noni yatara otoko-zuki meccha tarashide panko anna onna owatteru She’s just tall and apart from that she’s so freakin’ ugly, and despite that she’s such a cock-loving slut, she’s finished already.
Shinde kureeee, daibu kiraware-mono de yuumei, subete ga itaitashii... Please, dieeee, you’re so famous for being disliked by everyone, everything about you is so pathetic
Note: The upper three sentence pairs include strong sarcasm despite the overt positive expressions in the sentences. The English translation corresponds to Japanese original content. Harmful patterns recognized automatically are underlined (underlining in English corresponds as closely to the Japanese as possible).

Table 4. Three Examples of Preprocessing of a Sentence in Japanese

Sentence: 今日はなんて気持ちいい日なんだ！ • **Transcription in phonetic alphabet**: Kyōwanantekimochiiihinanda! • **Glosses**: Today TOP what pleasant day COP EXCL • **Translation**: What a pleasant day it is today!
Preprocessing Examples: • **Tokenization**: Kyō wa nante kimochiii hi nanda ! • **POS**: N PP ADV ADJ N AUX SYM • **Tokens+POS**: Kyō [N] wa[PP] nante[ADV] kimochi_ii[ADJ] hi[N] nanda[AUX] ![SYM] • **Chunks**: Kyō_wa nante kimochi_ii hi_nanda! • **Dependency relations**: *0_3D_Kyō_wa *1_2D_nante *2_3D_kimochi_ii *3_-1D_hi_nanda!
Notes: N = noun; PP = postpositional particle; ADV = adverb; ADJ = adjective; AUX = auxiliary verb; SYM = symbol; 1D, 2D, ... = depth of dependency relation; *0, *1, *2, ... = phrase number.

4.3 Experiment Setup

The preprocessed original data set provides nine separate training and test sets for the experiment (tokenized, POS-tagged, tokens with POS, lemmatized, lemmas with POS, chunks, dependency relations, chunks with named entities, dependency with named entities). The experiment was performed nine times, one time for each kind of preprocessing, in order to choose the best option. For each version of the data set, an experiment with a 10-fold cross-validation was performed and the results were calculated using standard Precision, Recall, balanced F-score, and Accuracy measures for each threshold within the whole threshold span. In one experiment 14 different versions of the proposed classifier were compared with the 10-fold cross-validation condition. Versions of the classifier represent combinations of weight calculation and pattern list modification explained in Section 3.3 and are, in order: PAT, PAT-0P, PAT-AMB, PAT-LA, PAT-LA-0P, PAT-LA-AMB, PAT-LOA, NGR, NGR-0P, NGR-AMB, NGR-LA, NGR-LA-0P, NGR-LA-AMB, NGR-LOA. Additionally, we performed experiments using classifiers applied in previous research using the tf*idf weight calculation schema. Since the experiment is performed for nine different versions of preprocessing, we obtained an overall number of 1980 experiment runs, 1260 for the proposed algorithm and an additional 720 for other classifiers (nine data sets * eight classifiers in 10-fold cross-validation). There were several evaluation criteria. First, we looked at which version of the algorithm achieved the highest balanced F-score, and the highest Accuracy within the threshold span. This is referred to as threshold optimization. However, theoretically, an algorithm could achieve its best score for one certain threshold, while for others it could perform poorly. Therefore, we also looked at break-even points (BEP) of Precision and Recall. This shows which version of the algorithm is more balanced. Finally, we checked the statistical significance of the results. We used a paired T-test because the classification results could represent only one of two classes (harmful or nonharmful). To choose the best version of the algorithm, we separately compared the results achieved by each group of modifications; e.g., "different pattern weight calculations," "pattern list modifications," and "patterns vs. n-grams." We also compared the performance to all applicable previous methods, which we considered as a baseline. This refers primarily to SO-PMI-IR-based methods (Matsuba et al., 2011; Nitta et al., 2013; Ptaszynski et al., 2016), which were evaluated on the same data, as well as the SVM-based classifier, which many previous studies selected as the best (e.g., Ptaszynski et al., 2010; Dinakar et al., 2012; Sood et al., 2012; Sarna & Bhatia 2017).

Table 5. Comparison of Best Precision, F-Score and Accuracy Within the Threshold Span for Each Version of The Classifier for All Data Sets.

Highest Precision within threshold					
		Pr	**Re**	**F1**	**Acc**
Tokenized	**(PAT-0P/NGR-0P)**	0.861	0.249	0.387	0.614
Lemmatized	**(PAT-LA-0P)**	0.902	0.208	0.338	0.602
Parts-or-Speech	**(NGR-LA)**	0.934	0.031	0.060	0.514
Tokens+POS	**(PAT-ALL/NGR-ALL)**	0.890	0.336	0.487	0.647
Lemmas+POS	**(PAT-AMB)**	0.956	0.119	0.212	0.567
Chunks	**(NGR-LA-0P)**	0.875	0.072	0.133	0.533
Dependency	**(PAT-LA)**	0.868	0.071	0.131	0.537
Chunks+NER	**(NGR-ALL)**	0.768	0.242	0.368	0.586
Dependency+NER	**(NGR-LA)**	0.718	0.010	0.020	0.513
Highest F-score within threshold					
		Pr	**Re**	**F1**	**Acc**
Tokenized	**(PAT-0P)**	0.724	0.842	0.778	0.766
Lemmatized	**(NGR-ALL)**	0.713	0.885	0.790	0.770
Parts-or-Speech	**(PAT-AMB)**	0.528	0.946	0.677	0.550
Tokens+POS	**(PAT-0P/NGR-0P)**	0.756	0.839	0.796	0.784
Lemmas+POS	**(NGR-ALL)**	0.807	0.798	0.803	0.808
Chunks	**(PAT-LA-0P/NGR-LA-0P)**	0.490	1.000	0.658	0.490
Dependency	**(PAT-LA/NGR-LA)**	0.491	1.000	0.658	0.491
Chunks+NER	**(NGR-ALL)**	0.563	0.879	0.686	0.603
Dependency+NER	**(NGR-0P)**	0.500	0.982	0.663	0.510
Highest Accuracy within threshold					
		Pr	**Re**	**F1**	**Acc**
Tokenized	**(PAT-AMB)**	0.778	0.760	0.769	0.776
Lemmatized	**(NGR-ALL)**	0.787	0.781	0.784	0.790
Parts-or-Speech	**(NGR-ALL)**	0.635	0.528	0.576	0.612
Tokens+POS	**(PAT-0P/NGR-0P)**	0.756	0.839	0.796	0.784
Lemmas+POS	**(NGR-ALL)**	0.807	0.798	0.803	0.808
Chunks	**(PAT-LA-0P)**	0.658	0.589	0.622	0.640
Dependency	**(PAT-LA-0P)**	0.671	0.336	0.448	0.580
Chunks+NER	**(NGR-ALL)**	0.659	0.647	0.653	0.655
Dependency+NER	**(NGR-0P)**	0.551	0.617	0.582	0.559

5 Results and Discussion

To obtain as objective a perspective as possible, we analyzed the results in two ways. First, we analyzed the results for each feature set separately in order to find out which weight calculation method and which pattern list modification achieved the highest results for each set. We looked at the highest F-score and Accuracy. We also checked the break-even point of Precision and Recall (BEP) for each version of the classifier. We also calculated statistical significance among all results within each feature set. Table 5 presents the best results (Precision, F-score, Accuracy) summarized for all data sets. Detailed separate results for best Precision, best F-score and best Accuracy for each version of the classifier, along with all T-tests are presented in the Appendix.

Second, we compared the results among all feature sets to find out whether there is a stable pattern in the results; e.g., if patterns are always better than n-grams, or if it is more effective to use all patterns, only the unambiguous ones, etc. Then, we analyzed whether the method works better on more generalized or more specific feature sets. Next, we calculated statistical significance between the best results of each feature set. Finally, we compared the best and the worst results of the proposed method to previous methods.

5.1 Results for Each Set Separately

5.1.1 Tokenized Data Set

At first, we looked at the data set preprocessed with the simplest method—namely, tokenization. Highest achieved Precision was 0.861 and was obtained by both patterns and n-grams when zero- patterns were deleted from pattern lists. The second- best Precision was achieved by the pattern list containing all patterns/n-grams (0.858). However, when all ambiguous patterns were deleted, the highest achieved Precision suddenly dropped to 0.820. Awarding length usually caused a drop in Precision.

The highest achieved F-score was 0.778 and was obtained by patterns when zero-patterns were deleted from pattern lists. The second-best F-score was achieved by the pattern list containing all patterns/n-grams (0.724). Deleting all ambiguous patterns caused a drop in the F-score to 0.690. Awarding length also caused a drop in the F-score.

The highest achieved Accuracy was 0.776 and was obtained by patterns when all ambiguous patterns were deleted from pattern lists. This stands in contradiction to the above results for best F-score, however, the second-best Accuracy was achieved by the pattern list containing all patterns/n-grams (0.766). Awarding length, similarly to above results, caused a drop in Accuracy.

In terms of statistical significance, differences between most results were statistically significant, meaning they could not be considered a matter of chance. Pairs that tended to be not statistically significant included those that differed only in one kind of characteristics, e.g., PAT-0P and NGR-0P, which were also two best scores for highest Precision and highest F-score (former).

5.1.2 Lemmatized Set

The second simplest way of preprocessing is lemmatization. In this process all declined and conjugated words are simplified to their dictionary forms. Therefore, lemmatization provides less specific (thus less differentiated) but more frequently appearing features. This makes lemmatization more generalized than tokenization.

The highest achieved Precision for the lemmatized data set was higher than for tokens and reached 0.902 when patterns were used with length-awarded weighting and deleted zero-patterns. In contrast to the tokenized data set, where different weighting and pattern list modifications caused a negative influence, for lemmas, the influence was, in most cases, positive, especially in terms of awarding length in the weight calculation.

The highest achieved F-score for the lemmatized data set was also higher than for the tokenized set and reached 0.79 for n-grams. The highest scores of pattern-based classifiers ranged lower in general. Like the tokenized data set, awarding length caused a drop in the highest achieved F-score. Also, deleting either zero-patterns or all ambiguous patterns/n-grams caused an occasional drop in scores, indicating that such patterns usually do not contribute positively to the classification based on lemmas.

Results for the highest Accuracy within the threshold confirm the above results for F-score. The highest result was achieved by n-grams and reached 0.79. Here, too, the pattern list and weighting modifications decreased the results, in general. The differences among all versions of the classifier were, in most cases, statistically significant, usually at the 0.1% level. Like the tokenized data set, the results, when not significant usually had only one difference, e.g., PAT-0P and NGR-0P, or NGR-AMB and NGR-LA-AMB. Unfortunately, the version of the classifier which achieved the highest top scores—namely, NGR-ALL (using all n-grams in classification)—did not reach statistical significance with pattern-based classifiers, which decreased the reliability of n-gram-based scores.

5.1.3 POS-Tagged Set

Compared to previous data sets, POS-tagging provides the most generalized way of preprocessing. A small number of features is extracted; however, they occur very frequently. Comparing the highest achieved scores, although the highest Precision score was 0.934 (Recall = 0.031), all other compared measures were generally much lower than for the tokenized and lemmatized data sets, reaching 0.677 and 0.612 at best, for F-score and Accuracy, respectively. This data set also reveals the highest number of cases in which differences between classifier versions were not statistically significant. There was also no consistency in terms of which version of the classifier performed best. This could suggest that the classifier performs poorly for highly generalized dataset with a small number of unique features, even if the occurrence of each feature is high.

5.1.4 Tokenized Set with POS

Data sets containing both tokens and POS information provide a feature set more specific than the original tokenized data set. Precision scores for this data set were also better, reaching 0.89 when either all patterns or all n-grams were used. Further modifications only diminished the results, although for the second-best Precision results, the corresponding F-score and Accuracy scores were much higher.

For the F-score as well as for Accuracy, the highest results were obtained when zero-patterns were deleted from the pattern list, reaching 0.796 and 0.784, respectively. Using all patterns, as well as deleting all ambiguous patterns diminished the results. Other modifications also failed to improve the results.

Unfortunately, more than half the T-test results indicated a lack of statistical significance. For the two best classifier versions (PAT-0P, NGR-0P), the differences were generally significant only in cases with no weight modifications.

5.1.5 Lemmatized Set with POS

The data set preprocessed to contain both lemmas and POS information (LEM+POS) is theoretically more generalized then TOK+POS, but more specific than either for tokenized only (TOK) or lemmatized only (LEM) data sets. The results for previous preprocessing means (TOK, LEM, POS) indicated that lemmas alone, although being more generalized than tokens, were more effective. On the other hand, parts of speech, though providing an even higher level of generalization, scored much lower. By combining lemmas with POS and comparing the results to TOK+POS we can evaluate whether POS could contribute positively to the overall results when they are combined with other features, even though they impose a negative influence when used alone. Indeed, based on the TOK+POS results, we could reasonably hypothesize that there might be some improvement.

The best Precision scores reached 0.956, which was the highest score for all previous results for any data set. This score was achieved by a pattern-based data set with all ambiguous patterns removed. However, both unmodified pattern- or n-gram-based classifiers also attained very high scores—0.954 and 0.948 respectively.

Results for both best F-score and best Accuracy revealed similar tendencies, with the n-gram-based unmodified classifier achieving F1=0.803 and A=0.808. These results confirm the positive influence of lemmas, as compared to simple tokens, and further supports the contribution of POS information, but only when it is also used with other features. T-test results for both F-score and Accuracy were, in most cases, either extremely statistically significant ($p \leq 0.001$) or very statistically significant ($p \leq 0.01$).

5.1.6 Chunk-Separated Set

While tokenization and lemmatization divide sentences according to words or morphemes, sentence parsing splits a sentence into unified meaningful chunks, typically consisting of more than one word. The simplest way of parsing is shallow parsing, or chunking, in which chunks are simply separated by spaces with no additional information provided about the relationships between the chunks. Chunking provides a more generalized preprocessing method than tokenization or lemmatization, but is less generalized then POS.

The highest Precision attained was 0.875, which is higher than for tokens alone, but lower than for lemmas or POS. Results for F-score and Accuracy were much lower than for all other sets (F1=0.658 and A=0.640). Also, the differences between the classifier versions were rarely statistically significant.

5.1.7 Dependency-Parsed Set

Dependency parsing of a sentence, or deep parsing, refers to splitting a sentence into chunks according to their syntactic interrelations. It is more specific than chunking; however, because of the small number of extracted frequent features available for building a reliable language model, its results are often inferior to those derived by simple tokenization unless really large data sets are in use.

The results of our experiment confirmed this. While the highest Precision scores reached 0.868, the best F-score and Accuracy scores were the worst of all data sets, reaching only F=0.658 and A=0.58, meaning that cyberbullying classification with the use of dependency parsing is about as accurate as random coin flipping. T-test results were also almost never statistically significant.

Table 6. Break-Even Points for all Feature Sets and All Classifier Versions.

	TOK	LEM	POS	TOK	LEM	CHUNK	DEP	CHUNK	DEP
				+POS	+POS			+NER	+NER
PAT	0.761	0.751	0.613	0.785	0.781	**0.633**	0.566	0.603	0.510
PAT-0P	0.763	0.751	0.613	**0.786**	0.781	0.632	0.551	0.605	0.512
PAT-AMB	**0.770**	0.751	0.613	0.764	0.782	0.629	**0.591**	0.603	0.514
PAT-LA	0.729	0.748	0.613	0.726	0.781	0.632	0.568	--	0.505
PAT-LA-0P	0.729	0.737	0.596	0.726	0.760	**0.633**	0.549	--	0.505
PAT-LA-AMB	0.711	0.737	0.594	0.715	0.761	0.629	**0.591**	--	0.516
NGR	0.761	**0.784**	**0.614**	0.785	**0.802**	0.632	0.566	**0.655**	0.547
NGR-0P	0.762	**0.784**	0.613	**0.786**	**0.802**	0.632	0.551	0.652	**0.548**
NGR-AMB	**0.770**	0.767	0.570	0.764	0.777	0.612	**0.591**	0.610	0.526
NGR-LA	0.729	0.767	0.605	0.726	0.762	**0.633**	0.551	0.619	0.546
NGR-LA-0P	0.729	0.768	0.607	0.726	0.769	0.631	0.559	0.622	**0.548**
NGR-LA-AMB	0.711	0.762	0.596	0.715	0.750	0.613	0.589	0.589	0.529

Note: Best scores for each classifier version underlined; best scores for each preprocessing method in **bold** font. No score means no BEP within the threshold range.

5.1.8 Chunk-Separated Set with Named Entities

Named entities, such as private names of people, companies, dates, numbers, etc., provide additional levels of generalization for analyzed sentences. We applied named-entity recognition (NER) to shallow and deep parsing to find out how such additional generalization would influence the results. The results were in general lower than for chunks alone, but higher than for dependency parsing. Much of the results were also statistically significant.

5.1.9 Dependency Parsed Set with Named Entities

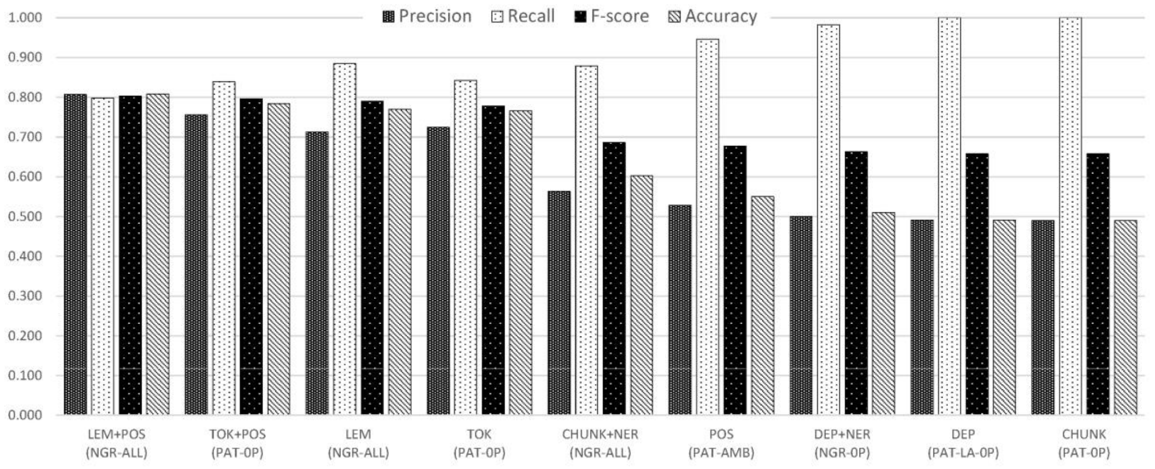
For dependency parsing supported with named entity recognition, the results were in general lower than for chunks with NER, but higher than for dependency parsing alone. The differences between the results were also more often statistically significant. This suggests that named entities, even for relatively small data sets, contribute positively to dependency parsing-based classification, although the contribution is still not strong enough to make the classification much better than random guessing.

5.2 Break-Even Point Analysis

Beyond the detailed analysis for each data set, we also looked at the results from a wider perspective. One of the popular methods of evaluation in text classification studies has been estimation of a break-even point (BEP), which is a cross-point of Precision and Recall, at which both of these scores (and the F-score result) are in equilibrium, meaning that the classifier is in its most balanced state. The higher the BEP, the more balanced the classifier. We calculated BEP for all versions of the proposed classifier (see Table 6 for results). Since versions with weight modified by awarding both pattern length and occurrence rarely obtained BEP within the threshold range, we excluded them from further analysis.

For the analysis we looked at four things: (1) which classifier version got the highest BEP, (2) which classifier version usually got the highest BEP for different data set preprocessing, (3) which preprocessing usually provided highest BEP, and (4) what was the highest achieved BEP and which data set / classifier combination produced this.

For the classifier version most frequently achieving top scores, although pattern-based and n-gram-based classifiers often achieved similar or even the same BEP scores, n-grams more frequently achieved the highest score. Moreover, the highest score of all was also achieved by the n-gram-based classifier, reaching P=R=F1=0.802.



Notes: Ordered according to highest F-score from left to right with corresponding precision, recall, and accuracy. The classifier version that achieved the score is in parentheses.

Figure 4. Best F-Scores for Each Dataset Preprocessing Method

Table 7. Analysis of Influence of Data Set Generalization on Results.

Data set Preprocessing			No. of unique unigrams	No. of all unigrams	Feature Density	Highest achieved F-score	Highest unmodified F-score	BEP
Feature sophistication	**high→**	**DEP**	12802	13957	0.917	0.658	0.658	0.591
		DEP+NER	12160	13956	0.871	0.663	0.662	0.548
		CHUNK	11389	13960	0.816	0.658	0.658	0.633
		CHUNK+NER	10657	13872	0.768	0.686	0.684	0.655
		TOK+POS	6565	34874	0.188	0.796	0.795	0.786
		TOK	6464	36234	0.178	0.778	0.778	0.770
	←low	**LEM+POS**	6227	36426	0.171	0.803	0.783	0.802
		LEM	6103	36412	0.168	0.790	0.764	0.784
		POS	13	26650	0.000	0.677	0.677	0.614
			unique 1ngr with			**FD with**		
			F1	**F1-unmod.**	**BEP**	**F1**	**F1-unmod.**	**BEP**
Pearson Correlation Coefficient (*ρ*-value)			-0.450 (p=0.224)	-0.453 (p=0.221)	-0.431 (p=0.247)	-0.735* (p=0.0242)	-0.736* (p=0.024)	-0.706* (p=0.0336)
with statistical significance (p-value)						**F1 & BEP**		**F1-unmod. & BEP**
(*p ≤ 0.05, ** p ≤ 0.01, ***p ≤ 0.001)						0.9681*** (p=0.00002)		0.9595*** (p=0.00004)

The data set preprocessing method that usually scored highest was the lemmatized data set combined with part-of-speech information. The highest score of all, mentioned above, was also achieved by this feature set.

Also, clustering the data set preprocessing methods into more generalized (POS, CHUNK, DEP, CHUNK+NER, DEP+NER, with BEP scores below 70%) and more specific (TOK, LEM, TOK+POS, LEM+POS, with BEP scores above 70%) methods, provides a meaningful insight. The classifier usually performs better on more specific feature sets. In other words, the method provides better results when it can extract large number of features, even if they occurred infrequently.

Although tokens with POS were the most numerous, the fact that lemmas with POS achieved the highest result needs to be addressed. Lemmatizing a sentence means that declined and conjugated forms of words are unified. This not only makes it easier to extract frequent patterns, but it is also advantageous in classification, since a pattern consisting of dictionary word forms applied to test data that has also been lemmatized provides much broader coverage. On the one hand, this offers the optimal setting for a classifier to use multiple specific features, but on the other hand, it must also be generalized to an extent so that it can capture a broad range of cases.

5.3 Comparison Between Feature Sets

Apart from analyzing BEPs, we also compared the highest achieved F-scores for each data set. These two evaluation measures rarely go together, since it would be difficult to have the highest F-score in place of the BEP. However, we can see which of the best F-scores is closest to the BEP. We can also follow the tendencies in the results (reported in Figure 4) to check if they correlate between the two evaluation measures. Lemmatization with POS information achieved the best score (F1=0.803 with P=0.807, R=0.798, and A=0.808). This confirms the winning setting for BEP, with a nearly identical F-score (0.802).

Interestingly, while the winning setting showed high consistency between Precision and Recall that was very close to BEP, for other data set preprocessing settings, the F-score was lower, and the gap between Precision and Recall was wider. This is meaningful not only because the F-score, and therefore the general performance of the data set was lower, but also because it provides further insight into the influence of generalization on results. Here, similarly to BEPs, the results can be clustered into two groups: one with a small gap between P and R, and another with a wide gap. This grouping is the same for BEP analysis.

Regarding the question of whether it is more useful to use pattern- or n-gram-based classifier, although the very best score was achieved by n-grams, both settings interchangeably appeared as the best settings. For example, the second-best setting was the pattern-based classifier using the pattern list with zero-patterns deleted, which achieved F1=0.796, P=0.756, R=0.839, and A=0.784. The third-best setting was the n-gram- based classifier, the fourth best was patterns once again, etc. This suggests that we need to perform more experiments, preferably using a wider threshold span to answer this question.

The optimal classifier setting was the unmodified one—or the one with zero-patterns deleted. This suggests, that, although ambiguous patterns appear both in cyberbullying as well as in normal messages, it is more effective to use them in classification. This is an interesting insight, since in previous research it has often been suggested that only words/patterns that are specifically characteristic to cyberbullying should be used. For example, as discussed above, Fujii et al. (2010) compared “gray words/patterns” with “black words/patterns” that appear only in harmful messages, opting to disregard gray words as noise.

5.4 Influence of Generalization on Results

To get a better grasp on the results, we also analyzed how the method used to preprocess a data set influenced the results.

To evaluate this we needed a quantifiable data set generalization measure. Broadly speaking, the more generalized a data set is, the fewer the number of frequently appearing unique features used in its preprocessing. Therefore, to estimate the data set generalization level, we decided to calculate feature sophistication level. We applied the lexical density (LD) score as the exact measure of feature sophistication level (Ure, 1971). LD is a score representing an estimated measure of content per lexical units for a given corpus and is calculated as the number of all unique words from a corpus divided by the number of all words in the corpus. However, because our research uses a variety of different features, not only words (tokens), and because the LD can vary depending on which features or feature combinations are used, we will henceforth call this measure feature-based lexical density, or *Feature Density* (FD), in short form.

After calculating FD for all used data sets we calculated Pearson’s correlation coefficient (ρ-value) to see if there was any correlation between data set generalization (FD) and the results. Pearson’s coefficient scores range from 1.0 (meaning there is a perfect positive correlation) through 0.0 (no correlation) to -1.0 (perfect negative correlation).

We used our highest F-scores as the results. However, because the highest overall F-scores were sometimes produced by different versions of the classifier (all patterns, or zero-patterns deleted; with length awarded, or without, etc.), we also used an unmodified version of the classifier (PAT-ALL). As an equivalent set of results, we also used BEPs. Finally, we verified whether the correlations were statistically significant.

While there was no correlation between unique unigrams of data sets and the overall results, the feature density score revealed an interesting correlation. There was a somewhat strong negative correlation (approximately -0.7) between the results and FD. This means that the results were better when the feature density was low. The correlation was not ideal, due to the fact that the preprocessing method resulting in the lowest FD (POS-tagged data set) produced some of the worst results. Interestingly, preprocessing methods resulting in very high FD (dependency parsing, etc.) also produced similarly poor results. This might suggest that there could be an even better preprocessing method yet to be discovered.

Table 8. Results of T-test for Best F-scores for Each Data Set.

	TOK+POS (PAT-0P)	LEM (NGR-ALL)	TOK (PAT-0P)	CHUNK+NER (NGR-ALL)	POS (PAT-AMB)	DEP+NER (NGR-0P)	DEP (PAT-LA-0P)	CHUNK (PAT-0P)
LEM+POS (NGR-ALL)	0.0005 ***	0.0284 *	0.0353 *	0.3062	0.0028 **	0.0878	0.0001 ***	0.1416
TOK+POS (PAT-0P)		0.7225	0.091	0.0035 **	0.0004 ***	0.4689	0.0001 ***	0.0001 ***
LEM (NGR-ALL)			0.0214 *	0.0102 *	0.0031 *	0.5168	0.0001 ***	0.0018 **
TOK (PAT-0P)				0.0576	0.0027 **	0.2423	0.0001 ***	0.0018 **
CHUNK+NER (NGR-ALL)					0.003 **	0.0329 *	0.0001 ***	0.0004 ***
POS (PAT-AMB)						0.002 **	0.0213 *	0.0091 **
DEP+NER (NGR-0P)							0.0001 ***	0.0013 **
DEP (PAT-LA-0P)								0.0001 ***

For the given data sets, the scores increase in tandem with decreasing FD, until the lowest FD is reached, which also resulted in low scores. Therefore, in the future we plan to use the FD measure to find a preprocessing method with optimal feature density, resulting in even better results. We present the analysis of the influence of data set generalization on the results in Table 7.

5.5 Statistical Significance of Results

As a final step in the analysis of the results for our proposed method, we analyzed the statistical significance of the highest F-scores for each data set preprocessing method. As a measure of significance, we used the Student's paired T-test, since the results represented either one of two sides (cyberbullying/harmful or nonharmful), and we compared all pairs of optimized methods (that produced the highest F-scores for the certain data set). Results of the T-test for the best F-scores for each data set are reported in Table 8.

We were most interested in how the best method (LEM+POS/NGR-ALL) compared to other methods, especially the worst methods. The worst methods (based on chunking, dependency parsing, and POS alone), always differed significantly with better methods (based on tokens, lemmas, and those combined with POS). Usually, the difference was very significant ($p \leq 0.01$) or even extremely significant ($p \leq 0.001$). There were some cases, however, in which the difference did not reach significance levels (e.g., CHUNK/PAT-0P vs. LEM+POS/NGR-ALL), indicating that we need to perform further experiments or widen the threshold span to obtain more result samples for comparison.

5.6 Comparison with Previous Methods

After specifying optimal settings for the proposed method, we compared it to previous methods. In the comparison, we used the methods proposed by Matsuba et al. (2011), Nitta et al. (2013), and, most recently, Ptaszynski et al. (2016). Moreover, since the Nitta et al.'s method extracts cyberbullying relevance values from the web (in particular Yahoo! API), beyond comparison to the reported results, we also repeated Nitta et al.'s experiment to find out how the performance of the web-based method has changed over the years. Finally, to make the comparison fairer, we compared both our best and worst results. For the evaluation metrics, we used the same one used in previous research—namely, area under the curve (AUC) on the graph in Figure 5 (which presents all results) showing Precision and Recall combined.

The highest overall results in terms of AUC were obtained by the best settings of the proposed method (based on all n-grams and normalized feature weight). Although the method starts low, it quickly attains over 95% of Precision and maintains this score even in the context of around 60% Recall.

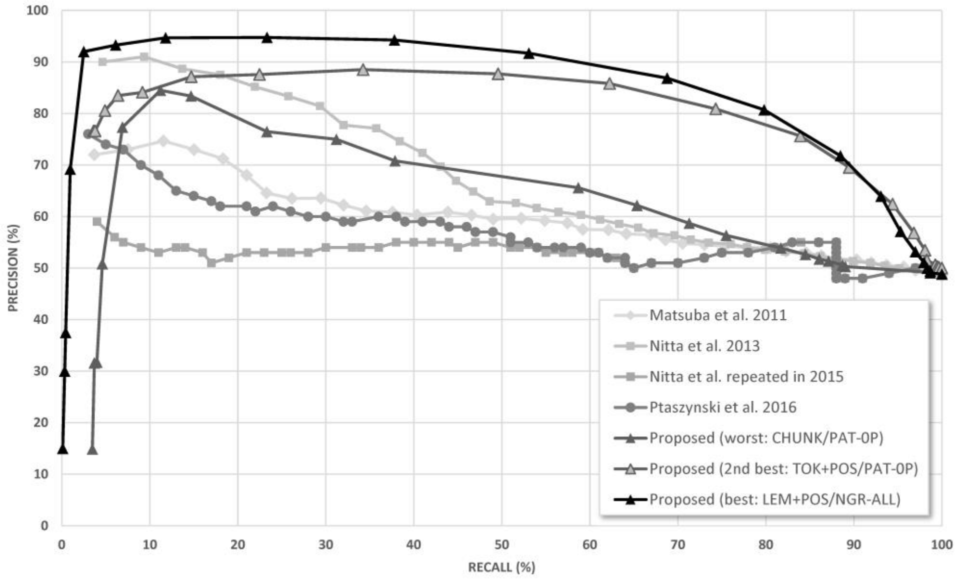


Figure 5. Comparison Between the Proposed Method (Best and Worst Performance) and Previous Methods.

Furthermore, even when the method loses Precision, the loss is not sudden or steep, but decreases slowly, maintaining high Precision values at around 70%, even when Recall reaches 90%. This method also outperformed all previous methods.

The second-best method (tokens with POS, all patterns, no weight modification) begins with a high Precision of 77% and retains it between 80% and 90% for most of the threshold. Although the highest originally reported Precision score by Nitta et al. (2013) was better than our second-best score, the performance of their method quickly decreased due to a quick drop in Precision for higher thresholds. Even our worst proposed method (based on separation by chunks with zero-patterns deleted) was still better than all other previous research, except for the original studies reported in 2013 and improved in 2016. Moreover, when we repeated Nitta et al.'s (2013) experiment in early 2015, the results declined greatly, losing over 30 percentage points of Precision. After a thorough analysis of the experiment data, we noticed that most of the information extracted in 2013 was not available in 2015. This could be due to the following reasons. First, fluctuation in page rankings of the applied search engine (Yahoo!) could have pushed the information further in the list making it inaccessible using Nitta et al.'s method. Second, frequent deletion requests of harmful contents by net-patrol members could potentially have had an effect. Third, recently, most web service providers have begun to tighten their usage policies. This applies to all major companies, including Google, Twitter, and Yahoo!, used by Nitta et al. (2013), and also applies to the recently introduced DMARC policies related to email spoofing and to general improvements in policies aimed at decreasing Internet harassment (http://www.dmarc.org). Such changes seeking to limit the growing problem of Internet harassment, implemented on a corporate level, are, in general, a positive phenomenon, despite reducing the performance of cyberbullying detection software. Moreover, as was recently shown by Ptaszynski et al. (2016), the performance of such software can, to some extent, be improved by automatically optimizing the list of seed words applied during use.

However, the fact that the performance of Nitta et al.'s method decreased from over 90% of Precision to less than 60% over three years is an important warning for other research based on web search-engine results. The probability of such problems was highlighted over a decade ago (Kilgarriff, 2007) and are likely to become a major problem in the future. This also suggests the need for more focus on corpus-based methods such as the one proposed here.

Finally, while our numerical results favored the proposed approach, we also wanted to understand the extent to which the patterns automatically recognized by the proposed method cover the manually selected seed words employed in previous research (Matsuba et al., 2010; Matsuba et al., 2011; Nitta et al., 2013). We found that all original seed words used in previous methods did, in fact, appear in the list of patterns automatically extracted by our proposed method. This can be interpreted as follows. First, the definition of cyberbullying (MEXT, 2008) and the intuition of researchers, which served as a primary basis for most previous approaches, were generally correct. Second, using our automatically extracted patterns, it may be possible to improve previous approaches in the future.

Finally, we also performed additional experiments using traditional classifiers applied in all previous research on automatic cyberbullying detection—namely, SVM, Naive Bayes, JRip, J48 and kNN, which we applied in experiments with the 10-fold cross-validation condition. All these classifiers consistently produced results that were inferior to the proposed method, optimized for each data set. Therefore, for each data set the winning classifier was always our proposed method.

The single case, in which a traditional classifier performed better than the proposed method, was the lemmatized data set classified with linear SVM on a bag-of-words with tf*idf weighting. However, since this classifier setting did not confirm its advantage over the proposed method for any other data set, it cannot be stated with any confidence that this is the optimal classification setting. The difference between the proposed best method and SVM-linear, when compared for all data sets, was statistically significant (p=0.0168, paired T-test). However, when the only one winning case for SVM was excluded, the significance was much higher (p=0.0044). This suggests the following interpretation. For all data sets, where the proposed method achieved higher results as compared to SVM-linear, the differences were very significant. The only case in which SVMs were superior, decreased the significance. Although we cannot discard this SVM classifier setting, due to its legitimately better results, in the future we will need to perform additional experiments in order to determine the definitive superior method.

As an additional remark, for traditional classifiers the tendencies in our results generally confirmed those produced by the proposed method. The results were better for tokenized and lemmatized data sets, with or without parts-of-speech, and much worse for POS-only, chunks, and dependency-parsed data sets. We present all results of experiments on traditional classifiers in Table 9.

Table 9. All Results of Experiments on Traditional Classifiers on All Data Sets.

			LEM +POS	TOK +POS	LEM	TOK	CHUNK +NER	POS	DEP +NER	DEP	CHUNK
SVM	**linear**	**Prec**	.777	.768	**.827**	.777	.679	.563	.651	.639	.606
		Rec	.777	.766	**.825**	.776	.645	.563	.615	.577	.603
		F1	.776	.766	**.825**	.775	.623	.563	.586	.531	.597
		Acc	.777	.766	**.825**	.776	.645	.563	.615	.577	.603
	plynomial	**Prec**	.262	.499	.262	.263	.260	.553	.260	.260	.260
		Rec	.512	.499	.512	.513	.510	.545	.510	.510	.510
		F1	.346	.450	.347	.348	.344	.528	.344	.344	.344
		Acc	.512	.499	.512	.513	.510	.545	.510	.510	.510
	radial	**Prec**	.797	.753	.262	.793	.260	.565	.260	.260	.260
		Rec	.771	.747	.512	.756	.510	.565	.510	.510	.510
		F1	.765	.746	.347	.746	.344	.565	.344	.344	.344
		Acc	.771	.747	.512	.756	.510	.565	.510	.510	.510
	sigmoid	**Prec**	.757	.746	.262	.752	.260	.562	.260	.260	.260
		Rec	.549	.736	.512	.538	.510	.562	.510	.510	.510
		F1	.425	.733	.347	.403	.344	.561	.344	.344	.344
		Acc	.549	.736	.512	.538	.510	.562	.510	.510	.510
Naive Bayes		**Prec**	.678	.671	.686	.682	.666	.570	.652	.672	.685
		Rec	.674	.669	.682	.678	.627	.569	.578	.555	.598
		F1	.673	.668	.681	.677	.599	.568	.511	.453	.539
		Acc	.674	.669	.682	.678	.627	.569	.578	.555	.598
JRip		**Prec**	.606	.614	.604	.603	.628	.553	.643	.505	.685
		Rec	.606	.613	.603	.603	.555	.553	.533	.510	.598
		F1	.606	.613	.603	.603	.469	.553	.408	.345	.539
		Acc	.606	.613	.603	.603	.555	.553	.533	.510	.598
J48		**Prec**	.672	.671	.683	.675	.615	.566	.652	.260	.645
		Rec	.671	.666	.681	.672	.548	.566	.533	.510	.517
		F1	.670	.663	.680	.669	.458	.566	.408	.344	.365
		Acc	.671	.666	.681	.672	.548	.566	.533	.510	.517
kNN (k=1)		**Prec**	.639	.630	.644	.630	.578	.544	.593	.628	.576
		Rec	.636	.627	.640	.628	.546	.543	.529	.528	.550
		F1	.633	.625	.637	.626	.505	.542	.446	.427	.494
		Acc	.636	.627	.640	.628	.546	.543	.529	.528	.550

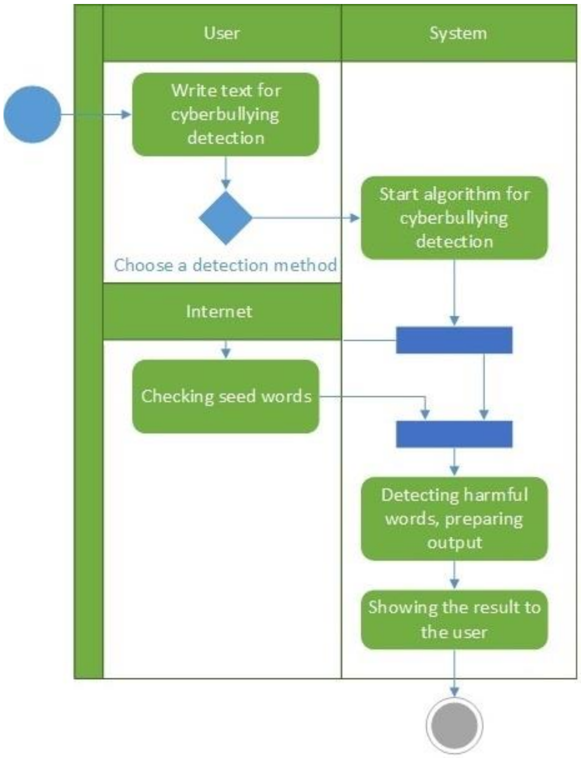

User
System
Write text for cyberbullying detection
Choose a detection method
Start algorithm for cyberbullying detection
Internet
Checking seed words
Detecting harmful words, preparing output
Showing the result to the user

Figure 6. Activity Diagram of the Application.

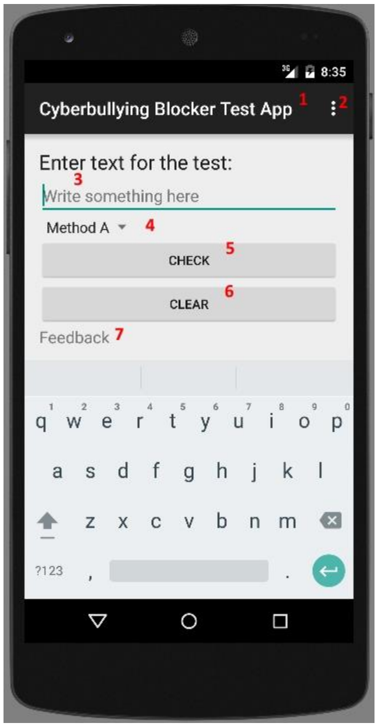


Figure 7. Interface of the Developed Application with Numbers (in Red) Marking Each Element of the Interface.

6 Cyberbullying Blocker Test Application for Android Devices

In this section, we present the mobile device application we developed to implement the proposed cyberbullying detection method. This application was created for devices supporting Android 4.2.2 (API level 17—i.e., “Jelly Bean”) or higher. In this process, Java 8 and Android Studio were used. The application contains one activity responsible for interaction with the user. For the process of checking the text for possible harmful content, it starts a background thread. Therefore, the user still can use the device even if the checking process takes a while. For comparison, beyond the cyberbullying detection method proposed in this paper, we also implemented Nitta et al.’s method (2013). We did this to test how the method proposed here would perform when compared to other methods. In the test phase, the user could use both methods to compare their overall performance, referring not only to Precision and Recall of detection, discussed in Section 4.5, but also to processing time and other features which could be used for further improvement of the proposed method.

Figure 6 represents an activity diagram of the detection process. Depending on the method of detection, the application may require an Internet connection. The application allows new methods for cyberbullying detection to be added or removed without affecting the operation of the application itself.

6.1 Description of Application Interface

The application interface is designed in accordance with the standards for the Android operating system[17] Interaction with a user is intuitive and essential information is accessible in the main window. Figure 7 represents the interface with a description of each important element. Below we describe the interface of the application. The numbers correspond to the numbers in red in Figure 7.

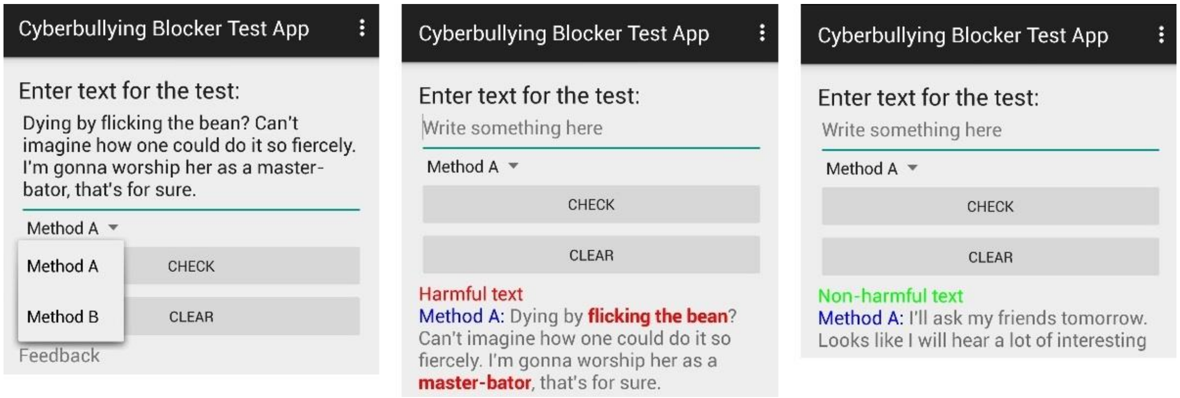
1. Name of the application
2. Settings, information about the application, and short descriptions of the methods used for cyberbullying detection
3. Input for text to classify
4. Choice of the method for cyberbullying classification
5. Button “check”: clicking on it starts the process of checking the entered text
6. Button “clear”: clears the input and feedback fields
7. Field showing the feedback of the method used for analyzing the text

6.2 Harmful Content Detection Process

Below we describe the process of detecting possible harmful content within the application.

1. **Sentence input.** First, the user inputs text to check. There is no limit on how long it can be, and the user can input several sentences at the same time. However, large inputs will slow the detection process for both methods. The only limitation is the phone memory for the TextView control class, which is usually set to 9,000 characters. Due to limitations of smartphone screen size, the application allows a maximum of five visible lines of input; to see additional lines the user must scroll down. The user can input any content desired, and the system will detect words and patterns recognizable by the used algorithm.
2. **Selection of detection method.** A user should choose one method from a list of all algorithms. Short description of methods can be found in the Settings tab of the application. The option of choosing the method is only available in the initial test version of the application and is intended to assist users in selecting the best algorithm for mobile devices.
3. **Launch of the checking process.**
4. **Feedback.** Processing the input text results in the selected method either detecting harmful patterns or not. In the first case, the feedback informs the user that no harmful words or patterns were detected in the input. The feedback also contains information about which classification method was used and the text entered by the user (see right part of Figure 8). In the second case, the feedback reports that the input sentence contained harmful content, identifies which method was used in detection, and shows the entered text and the detected harmful patterns in red bold font (see center section of Figure 8).

[17] http://developer.android.com/



Enter text for the test:

Dying by flicking the bean? Can't imagine how one could do it so fiercely. I'm gonna worship her as a master-bator, that's for sure.

Harmful text

Method A: Dying by **flicking the bean**? Can't imagine how one could do it so fiercely. I'm gonna worship her as a **master-bator**, that's for sure.

Non-harmful text

Method A: I'll ask my friends tomorrow. Looks like I will hear a lot of interesting

Figure 8. Text Input and Choice of Processing Method (Left), Harmful Text Output (Center), Nonharmful Text Output (Right).

6.3 Methods Description

For test purposes in the developed application, we plan to apply a number of different classification methods. In the first version of the application, described here, for the purpose of detection of harmful content in text on mobile devices, we used two previously developed methods of detection, both of which have been adapted to the Java language and Android environment. The methods can be replaced and updated in the future and we also plan on adding other methods.

6.3.1 Method A

As the first method ("Method A") we used the method presented in this paper, which classifies messages as harmful or not by using a classifier trained with a language modeling method based on a brute-force search algorithm applied to language modeling. Therefore, the efficiency of this method is essentially associated with the computing power of the device that is currently running the application. As all patterns used in classification are stored on the mobile device, the method can operate locally, and does not require an Internet connection.

6.3.2 Method B

The second method, further called ("Method B") uses a list of seed words from three categories to calculate the semantic orientation score SO-PMI-IR and then maximizes the relevance of categories using an input sentence according to a method developed by Nitta et al. (2013). The three steps in the classification of the harmfulness of input are:

1. Phrase extraction,
2. Categorization and harmful word detection together with harmfulness polarity determination,
3. Relevance maximization.

This method is an extension of the method proposed by Turney (2002) to calculate the relevance of words with specified categories according to the Equation 8, where p_i is a phrase extracted from the input, w_j are the words that are registered in one category of harmfulness polarity words, *hits*(p_i) and *hits*(w_j) are web search hits for each category for p_i and w_j respectively, *hits*(p_i&w_j) is a number of hits when p_i and w_j appear on the same web page. Finally, *PMI–IR*(p_i,w_j) is the relevance of p_i and w_j .

(8)

$$\mathrm{PMI-IR}(p_i, w_j) = log_2 \left\{ \frac{hits(p_i \& w_j)}{hits(p_i) hits(w_j)} \right\}$$

Turney's method was extended to work not only on words, but also on phrases. The phrases are automatically extracted from input sentences using dependency relations. Next, for all phrases, the relevance is calculated with seed words from multiple categories of harmful words. The degree of association for each category, the PMI-IR score, is maximized so that the maximum value achieved within all categories is considered as the harmfulness *score* representative for the input sentence and is calculated according to the Equation 9.

(9)

$$score_{entry} = \\ = max_{phrase}(max_{category}(PMI - IR(p_i, w_j)))$$

This method does not require excessive computer power and works on a small list of seed words, which can be further refined and optimized as shown by Ptaszynski et al. (2016). However, this method does require a stable Internet connection for calculating the PMI-IR score of the phrases with each group of seed words.

Table 10. Comparison of Different Features for Two Methods.

	Method A	Method B
Avg. # of words / patterns	11,832,430	9
used in classification		
Internet connection required	NO	YES
Avg. time for one sentence	88.46 s	8.41 s
(in seconds and minutes)	(1.47 m)	(0.14 m)

6.4 Preliminary Tests

We performed preliminary tests with the developed applications. The tests were not meant to check the validity of the applied methods, as this was already confirmed in previous papers (Nitta et al. 2013, Ptaszynski et al. 2016) and reconfirmed in this paper. We verified only whether the applied classification methods performed correctly under the new environment and whether they returned proper feedback.

For the purpose of testing the application, we prepared a set of sentences, derived from a data set used in previous research. Some of these sentences contained harmful words and some did not. We entered the sentences individually into the application input field and tested both methods. We present examples of outputs for harmful and nonharmful sentences in the center and right sections of Fig. 8, respectively. As our final goal, we plan to release the application for multiple languages. However, currently, since both of the applied cyberbullying detection methods only work using the Japanese language, the application was also developed for this language. However, for the purposes of this paper, we present the application interface and the sentence examples in English translation.

We performed tests on virtual devices emulated by Genymotion engine (Sony Xperia with Android 4.2.2 and Google Nexus 10 with Android 5.0) and on Smartphone LG G2 with Android 5.0.2. The tests focused on the performance of algorithms used on mobile devices, rather than on the usability of the application, because the current version of the application was created purely to verify if the detection algorithms work correctly on mobile devices and to evaluate which of them would be best for use in the full version of the application. Depending on the classification method, the detected harmful words may differ. Moreover, the processing speed of detection is associated with the type of device used (virtual smartphone, budget smartphone, or high-end smartphone) and the length of the text.

There was no clear winner. Method A achieves much better results and could be advantageous due to its lack of need for Internet connection. However, because it uses massive numbers of patterns, it takes about ten times longer to process one sentence than does Method B. On the other hand, while Method B works more quickly, it needs an Internet connection and the results are not as accurate. However, if the Method B could be improved so that it is as effective as Method A, it could be the better option, since access to Internet connection is an increasingly minor impediment. Results of the experiments are summarized in Table 10.

However, since Method A—the proposed method in this paper—produces far superior results, we believe that improving its processing speed, making it practical to implement on mobile devices, would be the best path forward. We have considered several ways to further this goal in the near future.

- **Distributed computing**. Since Method A classifies a massive number of sentence patterns, the processing is very time consuming. The processing time could be greatly reduced by distributing the pattern matching procedure on multiple CPU cores. Since modern mobile devices already use a multicore architecture, it may be possible to utilize it to improve the overall processing speed of running the method on such a device.
- **Feature filtering**. Another way to reduce processing time could involve filtering less useful patterns. Our experiments have already shown that in some situations deleting ambiguous patterns improved performance (F-score). This means that in such situations, not only is pattern matching performance improved, but also processing time, since deleting some patterns, in practice, equates to less time spent on processing individual sentences. Moreover, more sophisticated pattern filtering methods could be used. For example, increasingly popular deep learning methods also perform feature filtering in practice. Such methods do not scale well when a massive number of features is applied during the initial phase. However, this problem exists only during the training phase, which could be done on a powerful server. The final product would then contain only the patterns/features selected by the neural network during the training phase.

- **Applying more effective classifiers**. Along with feature filtering, it may be possible to apply a more time-efficient classifier. Since classifiers like SVMs, which also achieved comparatively good results in the experiments, are known for their fast processing time, it may be possible to apply them as the classifier in the proposed method.
- **Cloud API**. Finally, it may be possible to implement the method remotely, with a Cloud API. As such, the final product would only contain an I/O interface for the text message to process, while the processing itself would take place on a powerful remote server. While a constant Internet connection would be needed, this solution would completely remove the problem of insufficient hardware on the user side.

7 Conclusions and Future Work

In this paper, we proposed a method for automatic detection of Internet forum entries that contain cyberbullying messages; i.e., Internet content intended to humiliate and/or slander other people. Cyberbullying is an increasingly problematic social problem that negatively impacts the mental health of Internet users and has been implicated in self-harm and even suicide incidents among victims of cyberbullying.

The proposed method of counteracting cyberbullying applies a combinatorial algorithm, resembling a brute-force search algorithm, to automatically extract sophisticated sentence patterns, and uses those patterns for the text classification of cyberbullying entries. We tested this method on actual cyberbullying data obtained from the Human Rights Center, and performed experiments on nine different feature sets. The settings that were revealed as optimal consisted of word lemmas with part-of-speech information. Our results demonstrate that the proposed method outperformed all previously investigated methods. Moreover, when experiments from previous research were repeated, we found out that even the worst version of our proposed method outperformed previous methods, which declined in effectivity, in part, because they were primarily dependent on information extracted from the Internet. Apart from achieving superior performance, the proposed method is also more efficient than previous methods, as it requires minimal human effort.

We also presented an Android application for the detection of entries that contain cyberbullying. For the application, beyond the method proposed in this paper, we applied another method for comparison of features and performance. The main difference between the two detection methods from the point of view of software engineering, was that Method B (based on Nitta et al., (2013) required access to the Internet, while retaining low computing-power needs. Method A (proposed here), on the other hand, does not require Internet connection, but needs sufficient computing power. Since the future use of this application is inextricably linked with communication via the Internet, and each new generation of smartphones represents a major technological leap, we believe that these drawbacks will be quickly overcome by technological advances.

We outline two paths for further development. First, to improve the detection method itself, we plan to apply different means of data set preprocessing to find out whether performance can be further improved and to what extent. We found that too few highly generalized features (such as parts of speech alone) resulting in very low feature density, as well as too many overly specific features (sentence chunks with dependency relations) resulting in very high feature density, cause similarly poor results. In contrast, feature sets that are, to some extent, generalized but also plentiful (lemmas with POS), resulting in not-too-high and not-too-low feature density, produce the highest scores. We will pursue this path to discover the optimal feature density for the applied data set, and for the proposed method in general. We also plan to obtain new data to evaluate the method more thoroughly and plan to apply different classifiers. Finally, we plan to verify the actual amount of cyberbullying information on the Internet and reevaluate the method in more realistic conditions.

The second path for further development involves three main goals for the improvement of the developed smartphone application. First, we plan to evaluate how the performance of detection of the harmful words can be improved by using other methods or by optimizing the existing ones; in particular, for mobile devices. Second, we plan to identify the best way to implement our software on Internet communication applications such as Facebook or Twitter while conforming to all safety and privacy policies. At this point, we plan to either develop a plugin for existing applications or create a virtual keyboard for mobile devices. Finally, we plan to expand the scope of potential users by creating a version of the application for other dominant mobile systems (e.g., iOS).

References

Aramaki, E., Maskawa, S., & Morita, M. (2011). Twitter catches the flu: Detecting influenza epidemics using Twitter. *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 1568-1576).

Beijer, F. (2002). The syntax and pragmatics of exclamations and other expressive/emotional utterances (Working Papers in Linguistics 2, Lund University, Dept. of English). Retrieved from citeseerx.ist.psu.edu/viewdoc/download; jsessionid=255023532D85248C47ACE0F814 FC7C45?doi=10.1.1.13.5737&rep=rep1&type =pdf

Buntin, M. B., Burke, M. F., Hoaglin, M. C., & Blumenthal, D. (2011). The benefits of health information technology: A review of the recent literature shows predominantly positive results. *Health affairs*, *30*(3), 464-471.

Cambria, E. & Hussain, A. (2012). *Sentic Computing: Techniques, Tools, and Applications*. Dordrecht, Netherlands: Springer.

Cano, A. E., He, Y., Liu K. & Zhao J. (2013). A weakly supervised Bayesian model for violence detection in social media. *Proceedings of the 6th International Joint Conference on Natural Language Processing*.

Cross, D., Shaw, T., Hearn, L., Epstein, M., Monks, H., Lester, L., & Thomas, L. (2009). Australian covert bullying prevalence study. Retrieved from https://docs.education.gov.au/collections /australian-covert-bullying-prevalence-study.

Damashek, M. (1995). Gauging similarity with n-grams: Language independent categorization of text. *Science*, *267*(5199), 843.

Dinakar, K., Jones, B., Havasi, C., Lieberman, H., & Picard, R. (2012). Commonsense reasoning for detection, prevention and mitigation of cyberbullying, *ACM Transactions on Intelligent Interactive Systems*, *2*(3), Article 18.

Dooley J. J., Pyżalski J., & Cross D. (2009). Cyberbullying versus face-to- face bullying: A theoretical and conceptual review, *Zeitschrift für Psychologie / Journal of Psychology*, *217*(4), 182-188.

Fujii, Y., Ando, S., & Ito, T. (2010). *Yūgai jōhō firutaringu no tame no 2-tango-kan no kyori oyobi kyōki jōhō ni yoru bunshō bunrui shuhō no teian* [Developing a method based on 2-word co-occurrence information for filtering harmful information] (in Japanese), *Proceedings of The 24th Annual Conference of The Japanese Society for Artificial Intelligence* (pp. 1-4).

Gregor, S., & Hevner, A. (2013). Positioning and presenting design science research for maximum impact. *MIS Quarterly*, *37*(2), 337-355.

Guthrie, D., Allison, B., Liu, W., Guthrie, L., & Wilks, Y. (2006). A closer look at skip-gram modelling. *Proceedings of the 5th international Conference on Language Resources and Evaluation* (pp. 1-4).

Harris, Z. 1954. Distributional structure. *Word*, *10* (2/3), pp. 146-162.

Hasebrink, U., Görzig, A., Haddon, L., Kalmus, V., & Livingstone, S. (2011). *Patterns of risk and safety online: In-depth analyses from the EU Kids Online survey of 9-to 16-year-olds and their parents in 25 European countries*. London: EU Kids Online Network.

Hasebrink, U., Livingstone, S., Haddon, L., & O' lafsson, K. (2009). *Comparing children's online opportunities and risks across Europe: Cross-national comparisons for EU Kids Online*. London: EU Kids Online

Hashimoto, H., Kinoshita, T., & Harada, M. (2010). *Firutaringu no tame no ingo no yūgai goi kenshutsu kinō no imi kaiseki shisutemu SAGE e no kumikomi* [Implementing a function for filtering harmful slang words into the semantic analysis system SAGE] (in Japanese), *IPSJ SIG Notes*, 81(14), pp. 1-6.

Hevner, A., March, S., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *28*(1), pp. 75-105.

Hinduja, S., & Patchin, J. W. (2009). Bullying beyond the schoolyard: Preventing and responding to cyberbullying. Newbury Park, CA: Corwin.

Huang, X., Alleva, F., Hon, H.-W., Hwang, & M.-Y., Rosenfeld, R. (1992). The SPHINX-II speech recognition system: An overview, *Computer, Speech and Language*, *7*, pp. 137–148.

Ikeda, K., Yanagihara, T., Matsumoto, K., & Takishima, Y. (2010). *Kakuyōso no chūshōka ni motozuku ihō-, yūgai-bunsho kenshutsu shuhō no teian to hyōka* [Proposal and evaluation of a method for illegal and harmful document detection based on the abstraction of case elements] (in Japanese), *Proceedings of 72nd National Convention of Information Processing Society of Japan* (pp.71-72).

Ishisaka, T., & Yamamoto, K. (2010). *2channeru wo taishō to shita waruguchi hyōgen no chūshutsu* [Extraction of abusive expressions from

2channel] (in Japanese). *Proceedings of The Sixteenth Annual Meeting of The Association for Natural Language Processing* (pp.178-181).

Kann, L., Kinchen, S. Shanklin, S. L., Flint, K. H., Hawkins, J., Harris, … Zaza, S. (2013). Youth risk behavior surveillance: United states. morbidity and mortality weekly report. *Centers for Disease Control and Prevention*, *63*(4), 66.

Kilgarriff, A. (2007). Googleology is bad science. *Computational linguistics*, *33*(1), 147-151.

Kitajima, S., Rzepka, R., & Araki, K. (2014). Blog snippets based drug effects extraction system using lexical and grammatical restrictions. *International Journal of Multimedia Data Engineering and Management, 5*(2), 1-17.

Kontostathis, A., Reynolds, K., Garron, A., & Edwards, L. (2013). Detecting cyberbullying: Query terms and techniques. *Proceedings of the 5th ACM Web Science Conference* (195-204).

Kowalski, R. M., & Limber, S. P. (2007). Electronic bullying among middle school students. *Journal of Adolescent Health*, *41*(6), S22–S30.

Krippendorff, K. (1986). Combinatorial Explosion. *Web Dictionary of Cybernetics and Systems.* Retrieved from http://pespmc1.vub.ac.be/ASC/COMBIN_EXPLO.html

Lazuras L., Pyżalski J., Barkoukis V., & Tsorbazoudis H. (2012). Empathy and moral disengagement in adolescent cyberbullying: Implications for educational intervention and pedagogical Practice, *Studia Edukacyjne*, *23,* 57-69.

Leets, L. (2001). Responses to Internet hate sites: Is speech too free in cyberspace? *Communication Law and Policy*, *6*(2), 287-317.

Lau, R. Y., Liao, S. Y., Kwok, R. C. W., Xu, K., Xia, Y., & Li, Y. (2012). Text mining and probabilistic language modeling for online review spam detection. *ACM Transactions on. Management Information Systems*, *2*(4), Article 25.

Li, Q. (2007). Bullying in the new playground: Research into cyberbullying and cyber victimisation. *Australasian Journal of Education Technology, 23*(4), 435-454.

Marathe, S. S., & Shirsat, K. P. (2015). Contextual features based naïve Bayes classifier for cyberbullying detection on YouTube. *International Journal of Scientific and Engineering Research, 6*(11), 1109-1114.

Markov, A. A. (1971). Extension of the limit theorems of probability theory to a sum of variables connected in a chain. Reprinted in Appendix B of R. Howard. *Dynamic Probabilistic Systems* (Vol. 1: Markov Chains). New York, NY: Wiley.

Masui, F., Ptaszynski, M., & Nitta, T. (2013). *Intānetto-jō no yūgai kakikomi kenshutsu sōchi oyobi kenshutsu hōhō* [Device and method for detection of harmful entries on the Internet] (In Japanese). Patent application number 2013-245813.

Matsuba, T., Masui, F., Kawai, A., & Isu, N. (2010). *Gakkou hikoushiki saito ni okeru yuugai jouhou kenshutsu* [Detection of harmful information on informal school websites] (In Japanese). *Proceedings of the 16th Annual Meeting of The Association for Natural Language Processing.*

Matsuba, T., Masui, F., Kawai, A., & Isu, N. (2011). *Gakkō hi-kōshiki saito ni okeru yūgai jōhō kenshutsu wo mokuteki to shita kyokusei hantei moderu ni kansuru kenkyū* [A study on the polarity classification model for the purpose of detecting harmful information on informal school sites] (in Japanese). *Proceedings of The Seventeenth Annual Meeting of The Association for Natural Language Processing* (pp. 388-391).

Ministry of Education, Culture, Sports, Science and Technology (MEXT). (2008). 'Netto-jō no ijime' ni kansuru taiō manyuaru jirei shū (gakkō, kyōin muke) ["Bullying on the Net" Manual for handling and collection of cases (for schools and teachers)] (in Japanese). Retrieved from http://www.mext.go.jp/b_menu/houdou/20/11/08111701/001.pdf

Nitta, T., Masui, F. Ptaszynski, M. Kimura, Y., Rzepka, R., & Araki, K.. (2013). Detecting cyberbullying entries on informal school websites based on category relevance maximization. *Proceedings of the 6th International Joint Conference on Natural Language Processing* (pp. 579-586).

Patchin, J. W., & Hinduja, S. (2006). Bullies move beyond the schoolyard: A preliminary look at cyberbullying. *Youth Violence and Juvenile Justice*, *4*(2), 148-169.

Ponte, J. M., & Croft, W. B. (2017). A language modeling approach to information retrieval. In *ACM SIGIR Forum*, *51*(2), 202-208.

Potts, C., & Schwarz, F. (2008). Exclamatives and heightened emotion: Extracting pragmatic generalizations from large corpora (Unpublished manuscript). University of Massachusetts, Amherst, Massachusetts, USA.

Ptaszynski, M., Dybala, P., Matsuba, T., Masui, F., Rzepka, R., Araki, K., & Momouchi, Y. (2010).

In the service of online order: Tackling cyber-bullying with machine learning and affect analysis. *International Journal of Computational Linguistics Research*, *1*(3), 135-154.

Ptaszynski, M., Rzepka, R., Araki, K., & Momouchi, Y. (2011). Language combinatorics: A sentence pattern extraction architecture based on combinatorial explosion. *International Journal of Computational Linguistics*, *2*(1), 24-36.

Ptaszynski, M., Masui, F., Nitta, T., Hatakeyama, S., Kimura, Y., Rzepka, R., & Araki, K. (2016). Sustainable cyberbullying detection with category-maximized relevance of harmful phrases and double-filtered automatic optimization. *International Journal of Child-Computer Interaction*, *8*, 15-30.

Pyżalski, J. (2012). From cyberbullying to electronic aggression: typology of the phenomenon, *Emotional and Behavioural Difficulties*, *17*, 305-317

Sahlgren, M., & Cöster, R. (2004, August). Using bag-of-concepts to improve the performance of support vector machines in text categorization. *Proceedings of the 20th International Conference on Computational Linguistics* (p. 487).

Sarna, G., & Bhatia, M. P. S. (2017). Content based approach to find the credibility of user in social networks: an application of cyberbullying. *International Journal of Machine Learning and Cybernetics*, *8*(2), 677-689.

Sasai, K. (2006). The structure of modern Japanese exclamatory sentences: On the structure of the Nanto-type sentence. *Studies in the Japanese Language*, *2*(1), 16-31.

Siu, M., & Ostendorf, M. (2000). Variable n-grams and extensions for conversational speech language modeling. *IEEE Transactions on Speech and Audio Processing*, *8*(1), 63-75.

Smith, B., Ceusters, W., Klagges, B., Köhler, J., Kumar, A., Lomax, J., M., … Rosse, C., (2005). Relations in biomedical ontologies. *Genome biology*, *6*(5), R46.

Song, D., Huang, Q., Bruza, P., & Lau, R. (2012), An aspect query language model based on query decomposition and high-order contextual term associations. *Computational Intelligence*, *28*, 1-23.

Sood, S. O., Churchill, E. F., & Antin, J. (2012). Automatic identification of personal insults on social news sites. *Journal of the American Society for Information Science and Technology*, *63*(2), 270-285.

Sourander, A., Klomek, A. B., Ikonen, M., Lindroos, J., Luntamo, T., Koskelainen, M., ... Helenius, H. (2010). Psychosocial risk factors associated with cyberbullying among adolescents: A population-based study. *Archives of General Psychiatry*, *67*(7), 720-728.

Turney, P. D. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. *Proceedimgs of the 40th Annual Meeting of the Association for Computational Linguistics* (pp. 417-424).

Ure, J. (1971). Lexical density and register differentiation. In G. Perren and J.L.M. Trim (Eds.), *Applications of Linguistics* (pp. 443-452). London: Cambridge University Press.

Watanabe, H., & Sunayama, W. (2006). *Denshi keijiban ni okeru yūza no seishitsu no hyōka* [User nature evalution on BBS] (in Japanese). *IEICE Technical Report*, *105*(652), 25-30.

Appendix

Note: For all Appendix tables, best classifier version within each preprocessing kind is in bold.

Table A1. Comparison of Best Precision within the Threshold Span for Each Version of the Classifier for Tokenized Dataset.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.858	0.242	0.377	0.610
PAT-0P	**0.861**	**0.249**	**0.387**	**0.614**
PAT-AMB	0.820	0.491	0.614	0.699
PAT-LA-0P	0.838	0.150	0.255	0.571
PAT-LA	0.839	0.143	0.244	0.568
PAT-LA-AMB	0.755	0.562	0.644	0.697
NGR-ALL	0.859	0.243	0.378	0.611
NGR-0P	**0.861**	**0.249**	**0.387**	**0.614**
NGR-AMB	0.820	0.491	0.614	0.699
NGR-LA	0.840	0.144	0.245	0.568
NGR-LA-0P	0.838	0.150	0.254	0.570
NGR-LA-AMB	0.754	0.563	0.645	0.697

Note: Best classifier version within each preprocessing kind is in bold.

Table A2. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for Tokenized Dataset.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.724	0.842	0.778	0.766
PAT-0P	**0.724**	**0.842**	**0.778**	**0.766**
PAT-AMB	0.690	0.889	0.777	0.751
PAT-LA-0P	0.697	0.796	0.743	0.73
PAT-LA	0.656	0.856	0.742	0.709
PAT-LA-AMB	0.666	0.826	0.738	0.712
NGR-ALL	0.723	0.842	0.778	0.766
NGR-0P	0.724	0.841	0.778	0.765
NGR-AMB	0.690	0.889	0.777	0.751
NGR-LA	0.654	0.855	0.741	0.708
NGR-LA-0P	0.696	0.796	0.743	0.730
NGR-LA-AMB	0.666	0.826	0.737	0.711

Note: In case of identical results for f-score, the best score was optimized for accuracy**.**

Table A3. Comparison of best Accuracy Within the Threshold Span for Each Version of the Classifier for Tokenized Dataset.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.724	0.842	0.778	0.766
PAT-0P	**0.785**	**0.718**	**0.750**	**0.766**
PAT-AMB	0.778	0.760	0.769	0.776
PAT-LA-0P	0.736	0.713	0.724	0.734
PAT-LA	0.740	0.708	0.723	0.735
PAT-LA-AMB	0.715	0.703	0.709	0.718
NGR-ALL	0.723	0.842	0.778	0.766
NGR-0P	0.784	0.718	0.749	0.766
NGR-AMB	0.777	0.759	0.768	0.776
NGR-LA	0.740	0.708	0.723	0.735
NGR-LA-0P	0.737	0.713	0.725	0.735
NGR-LA-AMB	0.714	0.702	0.708	0.717

Note: In case of identical results for f-score, the best score was optimized for F-score.

Table A4. T-test Results (P-Values) for F-Scores and Accuracies Among all System Results for Tokenized Dataset.

F-score	PAT-	PAT-	PAT-	PAT-LA	PAT-	NGR-	NGR-	NGR-	NGR-LA	NGR-	NGR-	PAT-
	-ALL	-AMB	-LA	-AMB	-LA-0P	-ALL	-AMB	-LA	-AMB	-LA-0P	-0P	-0P
PAT-ALL		.000***	.002**	.002**	.002**	.920	.000***	.002**	.002**	.002**	.005**	.004**
PAT-AMB			.000***	.128	.000***	.000***	.068	.000***	.130	.000***	.000***	.000***
PAT-LA				.004**	.001***	.002**	.000***	.429	.004**	.001***	.003**	.003**
PAT-LA-AMB					.004**	.002**	.128	.004**	.023*	.004**	.002**	.002**
PAT-LA-0P						.002**	.000***	.001***	.004**	.147	.002**	.002**
NGR-ALL							.000***	.002**	.002**	.002**	.005**	.005**
NGR-AMB								.000***	.130	.000***	.000***	.000***
NGR-LA									.004**	.001***	.003**	.003**
NGR-LA-AMB										.004**	.002**	.002**
NGR-LA-0P											.002**	.002**
NGR-0P												.853
PAT-0P												
Accuracy												
PAT-ALL		.000***	.006**	.000***	.005**	.998	.000***	.006**	.000***	.005**	.001***	.001***
PAT-AMB			.000***	.272	.000***	.000***	.110	.000***	.265	.000***	.000***	.000***
PAT-LA				.000***	.000***	.006**	.000***	.697	.000***	.000***	.008**	.008**
PAT-LA-AMB					.000***	.000***	.268	.000***	.004**	.000***	.000***	.000***
PAT-LA-0P						.005**	.000***	.000***	.000***	.225	.006**	.006**
NGR-ALL							.000***	.006**	.000***	.004**	.001***	.001***
NGR-AMB								.000***	.261	.000***	.000***	.000***
NGR-LA									.000***	.000***	.009**	.009**
NGR-LA-AMB										.000***	.000***	.000***
NGR-LA-0P											.006**	.006**
NGR-0P												.802
PAT-0P												

$*p \leq 0.05$, $**p \leq 0.01$, $***p \leq 0.001$

Table A5. Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier for Lemmatized Dataset.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.871	0.276	0.419	0.627
PAT-0P	0.871	0.276	0.419	0.627
PAT-AMB	0.872	0.207	0.334	0.598
PAT-LA-0P	**0.902**	**0.208**	**0.338**	**0.602**
PAT-LA	0.872	0.293	0.438	0.633
PAT-LA-AMB	0.886	0.236	0.372	0.612
NGR-ALL	0.887	0.307	0.457	0.642
NGR-0P	0.886	0.329	0.48	0.651
NGR-AMB	0.810	0.597	0.687	0.735
NGR-LA	0.894	0.25	0.391	0.619
NGR-LA-0P	0.896	0.187	0.31	0.592
NGR-LA-AMB	0.807	0.609	0.694	0.738

Table A6. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for Lemmatized Dataset.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.715	0.82	0.764	0.753
PAT-0P	0.715	0.82	0.764	0.753
PAT-AMB	0.717	0.818	0.764	0.753
PAT-LA-0P	0.701	0.797	0.746	0.734
PAT-LA	0.714	0.811	0.76	0.749
PAT-LA-AMB	0.707	0.789	0.746	0.737
NGR-ALL	**0.713**	**0.885**	**0.79**	**0.77**
NGR-0P	0.713	0.885	0.79	0.769
NGR-AMB	0.713	0.864	0.781	0.763
NGR-LA	0.722	0.851	0.781	0.767
NGR-LA-0P	0.722	0.851	0.781	0.766
NGR-LA-AMB	0.724	0.818	0.768	0.758

Table A7. Comparison of Best Accuracy Within the Threshold Span for Each Version of the Classifier for Lemmatized Dataset.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.765	0.724	0.744	0.756
PAT-0P	0.765	0.724	0.744	0.756
PAT-AMB	0.765	0.724	0.744	0.756
PAT-LA-0P	0.748	0.718	0.732	0.744
PAT-LA	0.763	0.717	0.740	0.753
PAT-LA-AMB	0.749	0.717	0.733	0.745
NGR-ALL	**0.787**	**0.781**	**0.784**	**0.790**
NGR-0P	0.785	0.783	0.784	0.789
NGR-AMB	0.759	0.786	0.772	0.773
NGR-LA	0.770	0.762	0.766	0.773
NGR-LA-0P	0.769	0.766	0.768	0.773
NGR-LA-AMB	0.755	0.771	0.763	0.766
Note: Best classifier version within each preprocessing kind is in bold				

Table A8. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for Lemmatized Dataset.

F-score	PAT- -ALL	PAT- -AMB	PAT- -LA	PAT-LA -AMB	PAT- -LA-0P	NGR- -ALL	NGR- -AMB	NGR- -LA	NGR-LA -AMB	NGR- -LA-0P	NGR- -0P	PAT- -0P
PAT-ALL		.002**	.000***	.000***	.000***	.422	.001**	.001***	.001***	.001***	.222	.000***
PAT-AMB			.005**	.000***	.000***	.741	.001**	.001***	.001***	.001***	.421	.002**
PAT-LA				.000***	.000***	.803	.001**	.001**	.001***	.001**	.842	.000***
PAT-LA-AMB					.001***	.002**	.014*	.014*	.003**	.024*	.002**	.000***
PAT-LA-0P						.002**	.009**	.034*	.002**	.065	.003**	.000***
NGR-ALL							.002**	.001**	.001**	.001**	.005**	.422
NGR-AMB								.004**	.002**	.005**	.002**	.001**
NGR-LA									.002**	.001***	.001**	.001***
NGR-LA-AMB										.003**	.001**	.001***
NGR-LA-0P											.001**	.001***
NGR-0P												.222
PAT-0P												
Accuracy												
PAT-ALL		.000***	.000***	.000***	.000***	.310	.002**	.004**	.000***	.002**	.443	.000***
PAT-AMB			.003**	.000***	.000***	.202	.003**	.009**	.000***	.004**	.296	.000***
PAT-LA				.000***	.000***	.087	.007**	.045*	.000***	.020*	.135	.000***
PAT-LA-AMB					.000***	.000***	.165	.001***	.421	.001***	.000***	.000***
PAT-LA-0P						.000***	.283	.001**	.224	.002**	.000***	.000***
NGR-ALL							.000***	.000***	.000***	.000***	.000***	.310
NGR-AMB								.008**	.000***	.013*	.000***	.002**
NGR-LA									.000***	.000***	.000***	.004**
NGR-LA-AMB										.000***	.000***	.000***
NGR-LA-0P											.000***	.002**
NGR-0P												.443
PAT-0P												

$*p \leq 0.05, **p \leq 0.01, ***p \leq 0.001$

Table A9. Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier for POS-Tagged Dataset.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.776	0.048	0.091	0.518
PAT-0P	0.776	0.048	0.091	0.518
PAT-AMB	0.796	0.019	0.037	0.507
PAT-LA-0P	0.761	0.114	0.198	0.539
PAT-LA	0.776	0.048	0.091	0.518
PAT-LA-AMB	0.758	0.119	0.205	0.540
NGR-ALL	0.917	0.022	0.042	0.510
NGR-0P	0.917	0.021	0.041	0.510
NGR-AMB	0.652	0.196	0.302	0.546
NGR-LA	**0.934**	**0.031**	**0.060**	**0.514**
NGR-LA-0P	0.910	0.032	0.062	0.514
NGR-LA-AMB	0.656	0.308	0.419	0.574

Table A10. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for POS-Tagged Dataset.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.526	0.950	0.677	0.547
PAT-0P	0.526	0.950	0.677	0.547
PAT-AMB	**0.528**	**0.946**	**0.677**	**0.550**
PAT-LA-0P	0.524	0.952	0.676	0.543
PAT-LA	0.526	0.950	0.677	0.547
PAT-LA-AMB	0.526	0.949	0.677	0.547
NGR-ALL	0.518	0.959	0.672	0.533
NGR-0P	0.520	0.954	0.673	0.537
NGR-AMB	0.500	1.000	0.666	0.500
NGR-LA	0.528	0.935	0.675	0.551
NGR-LA-0P	0.530	0.930	0.675	0.552
NGR-LA-AMB	0.565	0.764	0.650	0.588

Table A11. Comparison of Best Accuracy Within the Threshold Span for Each Version of the Classifier for POS-Tagged Dataset.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.577	0.779	0.663	0.604
PAT-0P	0.577	0.779	0.663	0.604
PAT-AMB	0.578	0.776	0.663	0.605
PAT-LA-0P	0.639	0.461	0.536	0.600
PAT-LA	0.577	0.779	0.663	0.604
PAT-LA-AMB	0.640	0.463	0.537	0.600
NGR-ALL	**0.635**	**0.528**	**0.576**	**0.612**
NGR-0P	0.635	0.527	0.576	0.612
NGR-AMB	0.545	0.774	0.640	0.564
NGR-LA	0.624	0.539	0.578	0.607
NGR-LA-0P	0.626	0.540	0.580	0.609
NGR-LA-AMB	0.587	0.650	0.617	0.596

Table A12. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for POS-Tagged Dataset.

F-score	PAT-	PAT-	PAT-	PAT-LA	PAT-	NGR-	NGR-	NGR-	NGR-LA	NGR-	NGR-	PAT-
	-ALL	-AMB	-LA	-AMB	-LA-0P	-ALL	-AMB	-LA	-AMB	-LA-0P	-0P	-0P
PAT-ALL		.146	.000***	.009**	.009**	.086	.001***	.002**	.001**	.002**	.090	.000***
PAT-AMB			.146	.009**	.009**	.088	.001***	.002**	.001**	.002**	.093	.146
PAT-LA				.009**	.009**	.086	.001***	.002**	.001**	.002**	.090	.000***
PAT-LA-AMB					.076	.014*	.004**	.000***	.002**	.000***	.014*	.009**
PAT-LA-0P						.014*	.004**	.000***	.002**	.000***	.014*	.009**
NGR-ALL							.001**	.003**	.002**	.003**	.576	.086
NGR-AMB								.082	.005**	.092	.001**	.001***
NGR-LA									.014*	.007**	.003**	.002**
NGR-LA-AMB										.015*	.002**	.001**
NGR-LA-0P											.003**	.002**
NGR-0P												.090
PAT-0P												
Accuracy												
PAT-ALL		.065	.000***	.000***	.000***	.501	.000***	.000***	.000***	.000***	.474	.000***
PAT-AMB			.065	.000***	.000***	.656	.000***	.000***	.000***	.000***	.629	.065
PAT-LA				.000***	.000***	.501	.000***	.000***	.000***	.000***	.474	.000***
PAT-LA-AMB					.152	.000***	.043*	.628	.000***	.845	.000***	.000***
PAT-LA-0P						.000***	.052	.729	.000***	.965	.000***	.000***
NGR-ALL							.000***	.000***	.000***	.000***	.854	.501
NGR-AMB								.055	.000***	.040*	.000***	.000***
NGR-LA									.001***	.002**	.000***	.000***
NGR-LA-AMB										.001***	.000***	.000***
NGR-LA-0P											.000***	.000***
NGR-0P												.474
PAT-0P												

$*p \leq 0.05, **p \leq 0.01, ***p \leq 0.001$

Table A13. Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier for Dataset Containing Tokens with POS Information.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	**0.890**	**0.336**	**0.487**	**0.647**
PAT-0P	0.885	0.342	0.494	0.649
PAT-AMB	0.873	0.417	0.565	0.678
PAT-LA-0P	0.868	0.121	0.212	0.551
PAT-LA	0.868	0.092	0.167	0.539
PAT-LA-AMB	0.837	0.434	0.572	0.675
NGR-ALL	**0.890**	**0.336**	**0.487**	**0.647**
NGR-0P	0.886	0.344	0.496	0.650
NGR-AMB	0.873	0.417	0.565	0.678
NGR-LA	0.868	0.092	0.167	0.539
NGR-LA-0P	0.868	0.121	0.212	0.551
NGR-LA-AMB	0.837	0.434	0.572	0.675

Table A14. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for Dataset Containing Tokens with POS Information.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.754	0.840	0.795	0.783
PAT-0P	**0.756**	**0.839**	**0.796**	**0.784**
PAT-AMB	0.717	0.844	0.775	0.755
PAT-LA-0P	0.706	0.767	0.735	0.724
PAT-LA	0.700	0.775	0.736	0.722
PAT-LA-AMB	0.709	0.728	0.719	0.715
NGR-ALL	0.754	0.840	0.795	0.783
NGR-0P	**0.756**	**0.839**	**0.796**	**0.784**
NGR-AMB	0.717	0.844	0.775	0.755
NGR-LA	0.700	0.775	0.736	0.721
NGR-LA-0P	0.706	0.768	0.736	0.724
NGR-LA-AMB	0.709	0.728	0.719	0.715

Table A15. Comparison of Best Accuracy Within the Threshold Span for Each Version of the Classifier for Dataset Containing Tokens with POS Information.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.809	0.743	0.775	0.784
PAT-0P	**0.756**	**0.839**	**0.796**	**0.784**
PAT-AMB	0.741	0.809	0.774	0.763
PAT-LA-0P	0.736	0.708	0.722	0.727
PAT-LA	0.731	0.715	0.723	0.726
PAT-LA-AMB	0.718	0.710	0.714	0.715
NGR-ALL	0.809	0.744	0.775	0.784
NGR-0P	**0.756**	**0.839**	**0.796**	**0.784**
NGR-AMB	0.741	0.809	0.774	0.763
NGR-LA	0.731	0.715	0.723	0.726
NGR-LA-0P	0.735	0.707	0.721	0.726
NGR-LA-AMB	0.717	0.710	0.714	0.715

Table A16. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for Dataset Containing Tokens with POS Information.

F-score	PAT--ALL	PAT--AMB	PAT--LA	PAT-LA-AMB	PAT--LA-0P	NGR--ALL	NGR--AMB	NGR--LA	NGR-LA-AMB	NGR--LA-0P	NGR--0P	PAT--0P
PAT-ALL		.000***	.906	.128	.815	.583	.000***	.905	.128	.825	.001***	.001***
PAT-AMB			.000***	.005**	.000***	.000***	.530	.000***	.005**	.000***	.000***	.000***
PAT-LA				.031*	.005**	.905	.000***	.783	.031*	.018*	.956	.964
PAT-LA-AMB					.034*	.128	.005**	.031*	.315	.033*	.146	.145
PAT-LA-0P						.814	.000***	.005**	.034*	.218	.949	.941
NGR-ALL							.000***	.904	.128	.824	.001***	.001***
NGR-AMB								.000***	.005**	.000***	.000***	.000***
NGR-LA									.031*	.016*	.957	.965
NGR-LA-AMB										.033*	.146	.145
NGR-LA-0P											.958	.950
NGR-0P												.101
PAT-0P												
Accuracy												
PAT-ALL		.000***	.074	.010*	.064	.577	.000***	.074	.010*	.066	.001**	.001**
PAT-AMB			.000***	.486	.000***	.000***	1.000	.000***	.487	.000***	.000***	.000***
PAT-LA				.005**	.001***	.074	.000***	.834	.005**	.002**	.093	.091
PAT-LA-AMB					.006**	.010*	.486	.005**	.575	.006**	.013*	.013*
PAT-LA-0P						.064	.000***	.001***	.006**	.048*	.081	.080
NGR-ALL							.000***	.074	.010*	.065	.002**	.002**
NGR-AMB								.000***	.487	.000***	.000***	.000***
NGR-LA									.005**	.002**	.093	.092
NGR-LA-AMB										.006**	.013*	.013*
NGR-LA-0P											.082	.081
NGR-0P												.208
PAT-0P												

$*p \leq 0.05$, $**\,p \leq 0.01$, $***p \leq 0.001$

Table A17. Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier for Dataset Containing Lemmas with POS Information.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.954	0.114	0.204	0.565
PAT-0P	0.954	0.114	0.204	0.565
PAT-AMB	**0.956**	**0.119**	**0.212**	**0.567**
PAT-LA-0P	0.929	0.200	0.330	0.602
PAT-LA	0.954	0.114	0.204	0.565
PAT-LA-AMB	0.929	0.209	0.341	0.606
NGR-ALL	0.948	0.233	0.374	0.619
NGR-0P	0.948	0.119	0.212	0.567
NGR-AMB	0.932	0.205	0.336	0.604
NGR-LA	0.922	0.197	0.325	0.600
NGR-LA-0P	0.923	0.167	0.283	0.587
NGR-LA-AMB	0.892	0.266	0.409	0.626

Table A18. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for Dataset Containing Lemmas with POS Information.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.752	0.817	0.783	0.779
PAT-0P	0.752	0.817	0.783	0.779
PAT-AMB	0.759	0.813	0.785	0.782
PAT-LA-0P	0.725	0.809	0.765	0.756
PAT-LA	0.752	0.817	0.783	0.779
PAT-LA-AMB	0.732	0.805	0.766	0.760
NGR-ALL	**0.807**	**0.798**	**0.803**	**0.808**
NGR-0P	0.808	0.795	0.802	0.808
NGR-AMB	0.754	0.809	0.781	0.778
NGR-LA	0.733	0.794	0.763	0.756
NGR-LA-0P	0.720	0.825	0.769	0.757
NGR-LA-AMB	0.729	0.800	0.763	0.757

Table A19. Comparison of Best Accuracy Within the Threshold Span for each Version of the Classifier for Dataset Containing Lemmas with POS Information.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.817	0.737	0.775	0.791
PAT-0P	0.817	0.737	0.775	0.791
PAT-AMB	0.818	0.733	0.773	0.790
PAT-LA-0P	0.757	0.765	0.761	0.765
PAT-LA	0.817	0.737	0.775	0.791
PAT-LA-AMB	0.762	0.759	0.761	0.766
NGR-ALL	**0.807**	**0.798**	**0.803**	**0.808**
NGR-0P	0.808	0.795	0.802	0.808
NGR-AMB	0.788	0.764	0.776	0.784
NGR-LA	0.783	0.738	0.760	0.770
NGR-LA-0P	0.776	0.760	0.768	0.775
NGR-LA-AMB	0.729	0.800	0.763	0.757

Table A20. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for Lemmatized Dataset with POS Information.

F-score	PAT- -ALL	PAT- -AMB	PAT- -LA	PAT-LA -AMB	PAT- -LA-0P	NGR- -ALL	NGR- -AMB	NGR- -LA	NGR-LA -AMB	NGR- -LA-0P	NGR- -0P	PAT- -0P
PAT-ALL		.000***	.000***	.001***	.001***	.034*	.000***	.001***	.000***	.001***	.048*	.000***
PAT-AMB			.000***	.001***	.001***	.024*	.000***	.001**	.001***	.002**	.032*	.000***
PAT-LA				.001***	.001***	.034*	.000***	.001***	.000***	.001***	.048*	.000***
PAT-LA-AMB					.002**	.001***	.282	.001***	.000***	.001***	.001**	.001***
PAT-LA-0P						.001***	.340	.001***	.000***	.001***	.001**	.001***
NGR-ALL							.001***	.001**	.001***	.002**	.001**	.034*
NGR-AMB								.058	.038*	.002**	.001***	.000***
NGR-LA									.000***	.001***	.002**	.001***
NGR-LA-AMB										.000***	.001***	.000***
NGR-LA-0P											.003**	.001***
NGR-0P												.048*
PAT-0P												
Accuracy												
PAT-ALL		.002**	.000***	.000***	.000***	.018*	.000***	.000***	.000***	.001**	.027*	.000***
PAT-AMB			.002**	.000***	.000***	.012*	.000***	.001***	.000***	.002**	.016*	.002**
PAT-LA				.000***	.000***	.018*	.000***	.000***	.000***	.001**	.027*	.000***
PAT-LA-AMB					.000***	.000***	.098	.000***	.000***	.000***	.000***	.000***
PAT-LA-0P						.000***	.128	.000***	.000***	.000***	.000***	.000***
NGR-ALL							.000***	.000***	.000***	.001***	.003**	.018*
NGR-AMB								.010*	.019*	.000***	.000***	.000***
NGR-LA									.000***	.000***	.000***	.000***
NGR-LA-AMB										.000***	.000***	.000***
NGR-LA-0P											.001**	.001**
NGR-0P												.027*
PAT-0P												

$*p \leq 0.05, **p \leq 0.01, ***p \leq 0.001$

Table A21 Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier for Chunk-Separated Dataset.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.852	0.109	0.194	0.548
PAT-0P	0.845	0.112	0.198	0.548
PAT-AMB	0.849	0.087	0.158	0.534
PAT-LA-0P	0.845	0.115	0.202	0.549
PAT-LA	0.851	0.110	0.195	0.548
PAT-LA-AMB	0.850	0.088	0.160	0.535
NGR-ALL	0.848	0.110	0.195	0.547
NGR-0P	0.873	0.071	0.131	0.532
NGR-AMB	0.808	0.107	0.188	0.541
NGR-LA	0.856	0.111	0.197	0.549
NGR-LA-0P	**0.875**	**0.072**	**0.133**	**0.533**
NGR-LA-AMB	0.819	0.106	0.188	0.541

Table A22. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for Chunk-Separated Dataset.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.490	1.000	0.658	0.490
PAT-0P	0.490	1.000	0.658	0.490
PAT-AMB	0.490	1.000	0.658	0.490
PAT-LA-0P	**0.490**	**1.000**	**0.658**	**0.490**
PAT-LA	0.490	1.000	0.658	0.490
PAT-LA-AMB	0.490	0.999	0.658	0.490
NGR-ALL	0.490	1.000	0.658	0.490
NGR-0P	0.490	1.000	0.658	0.490
NGR-AMB	0.490	1.000	0.658	0.490
NGR-LA	**0.490**	**1.000**	**0.658**	**0.490**
NGR-LA-0P	**0.490**	**1.000**	**0.658**	**0.490**
NGR-LA-AMB	0.490	0.999	0.658	0.490

Table A23. Comparison of Best Accuracy Within the Threshold Span for Each Version of the Classifier for Chunk-Separated Dataset.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.656	0.587	0.620	0.638
PAT-0P	0.656	0.587	0.620	0.638
PAT-AMB	0.644	0.604	0.624	0.631
PAT-LA-0P	**0.658**	**0.589**	**0.622**	**0.640**
PAT-LA	0.655	0.587	0.619	0.638
PAT-LA-AMB	0.645	0.605	0.625	0.631
NGR-ALL	0.655	0.586	0.619	0.638
NGR-0P	0.657	0.586	0.620	0.639
NGR-AMB	0.606	0.635	0.620	0.613
NGR-LA	0.656	0.589	0.621	0.639
NGR-LA-0P	0.657	0.587	0.620	0.639
NGR-LA-AMB	0.606	0.637	0.621	0.613

Table A24. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for Chunk-Separated Dataset.

F-score	PAT-	PAT-	PAT-	PAT-LA	PAT-	NGR-	NGR-	NGR-	NGR-LA	NGR-	NGR-	PAT-
	-ALL	-AMB	-LA	-AMB	-LA-0P	-ALL	-AMB	-LA	-AMB	-LA-0P	-0P	-0P
PAT-ALL		.006**	.084	.005**	.002**	.187	.030*	.035*	.015*	.005**	.109	.002**
PAT-AMB			.007**	.045*	.015*	.005**	.215	.008**	.048*	.014*	.007**	.009**
PAT-LA				.005**	.002**	.003**	.034*	.083	.017*	.008**	.227	.012*
PAT-LA-AMB					.010**	.004**	.413	.006**	.079	.009**	.006**	.007**
PAT-LA-0P						.001***	.056	.002**	.026*	.255	.048*	.007**
NGR-ALL							.026*	.000***	.013*	.003**	.051	.002**
NGR-AMB								.036*	.044*	.055	.031*	.042*
NGR-LA									.018*	.011*	.372	.071
NGR-LA-AMB										.025*	.017*	.021*
NGR-LA-0P											.009**	.025*
NGR-0P												.940
PAT-0P												
Accuracy												
PAT-ALL		.098	.342	.057	.000***	.000***	.126	.004**	.102	.000***	.002**	.000***
PAT-AMB			.113	.044*	.857	.026*	.151	.229	.111	.349	.986	.459
PAT-LA				.066	.000***	.000***	.132	.025*	.107	.000***	.003**	.000***
PAT-LA-AMB					.582	.015*	.199	.134	.147	.645	.696	.290
PAT-LA-0P						.000***	.277	.001***	.232	.012*	.641	.011*
NGR-ALL							.084	.000***	.067	.000***	.000***	.000***
NGR-AMB								.165	.117	.392	.276	.210
NGR-LA									.135	.001***	.017*	.063
NGR-LA-AMB										.331	.231	.173
NGR-LA-0P											.005**	.001***
NGR-0P												.058
PAT-0P												

$*p \leq 0.05, **p \leq 0.01, ***p \leq 0.001$

Table A25. Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier with Dependency Parsed Dataset.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.865	0.070	0.130	0.537
PAT-0P	0.865	0.070	0.130	0.537
PAT-AMB	0.860	0.069	0.129	0.537
PAT-LA-0P	0.866	0.071	0.131	0.537
PAT-LA	**0.868**	**0.071**	**0.131**	**0.537**
PAT-LA-AMB	0.860	0.069	0.129	0.537
NGR-ALL	0.865	0.071	0.130	0.537
NGR-0P	0.850	0.070	0.129	0.537
NGR-AMB	0.860	0.069	0.129	0.537
NGR-LA	0.865	0.070	0.130	0.537
NGR-LA-0P	0.862	0.069	0.129	0.537
NGR-LA-AMB	0.856	0.069	0.128	0.537

Table A26. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier with Dependency Parsed Dataset.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.490	1.000	0.658	0.490
PAT-0P	0.490	1.000	0.658	0.490
PAT-AMB	0.490	1.000	0.658	0.490
PAT-LA-0P	**0.491**	**1.000**	**0.658**	**0.491**
PAT-LA	**0.491**	**1.000**	**0.658**	**0.491**
PAT-LA-AMB	**0.491**	**1.000**	**0.658**	**0.491**
NGR-ALL	0.490	1.000	0.658	0.490
NGR-0P	0.490	1.000	0.658	0.490
NGR-AMB	0.490	1.000	0.658	0.490
NGR-LA	**0.491**	**1.000**	**0.658**	**0.491**
NGR-LA-0P	**0.491**	**1.000**	**0.658**	**0.491**
NGR-LA-AMB	**0.491**	**1.000**	**0.658**	**0.491**

Table A27. Comparison of Best Accuracy Within the Threshold Span for Each Version of the Classifier with Dependency Parsed Dataset.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.671	0.336	0.448	0.580
PAT-0P	0.671	0.336	0.448	0.580
PAT-AMB	0.659	0.340	0.448	0.574
PAT-LA-0P	**0.671**	**0.336**	**0.448**	**0.580**
PAT-LA	0.670	0.335	0.447	0.579
PAT-LA-AMB	0.659	0.340	0.449	0.574
NGR-ALL	0.671	0.336	0.448	0.580
NGR-0P	0.672	0.335	0.447	0.580
NGR-AMB	0.659	0.340	0.449	0.574
NGR-LA	0.671	0.336	0.448	0.580
NGR-LA-0P	0.672	0.335	0.448	0.580
NGR-LA-AMB	0.659	0.340	0.449	0.574

Table A28. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for Dependency Parsed Dataset.

F-score	PAT- -ALL	PAT- -AMB	PAT- -LA	PAT-LA -AMB	PAT- -LA-0P	NGR- -ALL	NGR- -AMB	NGR- -LA	NGR-LA -AMB	NGR- -LA-0P	NGR- -0P	PAT- -0P
PAT-ALL		.176	.576	.238	.969	.000***	.184	.211	.180	.010**	.000***	.023*
PAT-AMB			.136	.057	.160	.108	.177	.154	.492	.240	.242	.219
PAT-LA				.186	.291	.186	.143	.862	.139	.016*	.025*	.062
PAT-LA-AMB					.218	.152	.221	.209	.000***	.318	.321	.294
PAT-LA-0P						.014*	.168	.462	.163	.011*	.019*	.045*
NGR-ALL							.113	.020*	.113	.000***	.000***	.000***
NGR-AMB								.161	.253	.253	.255	.231
NGR-LA									.157	.000***	.000***	.010*
NGR-LA-AMB										.240	.244	.222
NGR-LA-0P											.814	.533
NGR-0P												.354
PAT-0P												
Accuracy												
PAT-ALL		.600	.998	.516	.007**	.000***	.529	.149	.568	.036*	.136	.029*
PAT-AMB			.598	.000***	.818	.653	.003**	.631	.187	.790	.714	.673
PAT-LA				.514	.023*	.190	.528	.351	.566	.081	.260	.230
PAT-LA-AMB					.713	.566	.704	.543	.000***	.685	.615	.581
PAT-LA-0P						.048*	.728	.010*	.780	.403	.018*	.005**
NGR-ALL							.580	.591	.620	.142	.405	.475
NGR-AMB								.557	.165	.700	.629	.594
NGR-LA									.598	.058	.270	.235
NGR-LA-AMB										.751	.677	.638
NGR-LA-0P											.015*	.053
NGR-0P												.360
PAT-0P												

$*p \leq 0.05, **\, p \leq 0.01, ***p \leq 0.001$

Table A29. Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier for Chunks with NER.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.637	0.165	0.262	0.546
PAT-0P	0.637	0.167	0.265	0.546
PAT-AMB	0.644	0.199	0.304	0.551
PAT-LA-0P	0.540	0.783	0.639	0.545
PAT-LA	0.592	0.698	0.641	0.543
PAT-LA-AMB	0.546	0.761	0.635	0.542
NGR-ALL	**0.768**	**0.242**	**0.368**	**0.586**
NGR-0P	0.749	0.251	0.376	0.585
NGR-AMB	0.724	0.186	0.297	0.558
NGR-LA	0.736	0.252	0.375	0.582
NGR-LA-0P	0.730	0.270	0.395	0.585
NGR-LA-AMB	0.644	0.209	0.315	0.552

Table A30. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for Chunks with NER.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.551	0.902	0.684	0.575
PAT-0P	0.552	0.903	0.685	0.577
PAT-AMB	0.541	0.913	0.680	0.565
PAT-LA-0P	0.508	0.964	0.666	0.511
PAT-LA	0.507	0.984	0.669	0.515
PAT-LA-AMB	0.508	0.967	0.666	0.513
NGR-ALL	**0.563**	**0.879**	**0.686**	**0.603**
NGR-0P	0.560	0.864	0.680	0.599
NGR-AMB	0.554	0.851	0.671	0.587
NGR-LA	0.579	0.820	0.678	0.616
NGR-LA-0P	0.546	0.904	0.681	0.583
NGR-LA-AMB	0.523	0.906	0.664	0.548

Table 41. Comparison of Best Accuracy Within the Threshold Span for Each Version of the Classifier for Chunks with NER.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.596	0.668	0.630	0.602
PAT-0P	0.598	0.670	0.632	0.603
PAT-AMB	0.593	0.708	0.645	0.602
PAT-LA-0P	0.540	0.783	0.639	0.545
PAT-LA	0.592	0.698	0.641	0.543
PAT-LA-AMB	0.546	0.761	0.635	0.542
NGR-ALL	**0.659**	**0.647**	**0.653**	**0.655**
NGR-0P	0.655	0.646	0.650	0.651
NGR-AMB	0.622	0.568	0.594	0.613
NGR-LA	0.625	0.596	0.610	0.623
NGR-LA-0P	0.626	0.605	0.616	0.625
NGR-LA-AMB	0.588	0.663	0.624	0.603

Table A32. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for Dataset Separated By Chunks with NER.

F-score	PAT- -ALL	PAT- -AMB	PAT- -LA	PAT-LA -AMB	PAT- -LA-0P	NGR- -ALL	NGR- -AMB	NGR- -LA	NGR-LA -AMB	NGR- -LA-0P	NGR- -0P	PAT- -0P
PAT-ALL		.006**	.021*	.021*	.021*	.017*	.051	.153	.127	.599	.017*	.016*
PAT-AMB			.026*	.026*	.025*	.010*	.018*	.020*	.339	.073	.010**	.008**
PAT-LA				.047*	.088	.004**	.003**	.009**	.009**	.012*	.004**	.021*
PAT-LA-AMB					.088	.004**	.003**	.010**	.010**	.013*	.004**	.022*
PAT-LA-0P						.004**	.003**	.010**	.009**	.012*	.004**	.021*
NGR-ALL							.027*	.008**	.005**	.005**	.046*	.015*
NGR-AMB								.026*	.002**	.008**	.027*	.041*
NGR-LA									.013*	.000***	.007**	.099
NGR-LA-AMB										.030*	.004**	.142
NGR-LA-0P											.004**	.438
NGR-0P												.015*
PAT-0P												
Accuracy												
PAT-ALL		.911	.029*	.026*	.031*	.031*	.010**	.002**	.000***	.000***	.028*	.046*
PAT-AMB			.023*	.021*	.025*	.033*	.011*	.003**	.000***	.000***	.031*	.559
PAT-LA				.028*	.063	.003**	.000***	.002**	.000***	.001***	.003**	.028*
PAT-LA-AMB					.214	.003**	.001***	.003**	.000***	.001***	.003**	.025*
PAT-LA-0P						.004**	.001***	.003**	.000***	.001***	.003**	.030*
NGR-ALL							.462	.334	.392	.863	.374	.034*
NGR-AMB								.520	.456	.659	.497	.012*
NGR-LA									.866	.000***	.345	.002**
NGR-LA-AMB										.089	.403	.000***
NGR-LA-0P											.921	.000***
NGR-0P												.031*
PAT-0P												

$*p \leq 0.05, **p \leq 0.01, ***p \leq 0.001$

Table A33. Comparison of Best Precision Within the Threshold Span for Each Version of the Classifier for Dependency Parsed Set with NER.

Highest Precision within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.529	0.160	0.246	0.513
PAT-0P	0.547	0.167	0.256	0.516
PAT-AMB	0.611	0.185	0.284	0.531
PAT-LA-0P	0.513	0.805	0.627	0.529
PAT-LA	0.513	0.805	0.627	0.528
PAT-LA-AMB	0.518	0.507	0.513	0.527
NGR-ALL	0.699	0.100	0.174	0.521
NGR-0P	0.688	0.101	0.176	0.520
NGR-AMB	0.619	0.216	0.320	0.536
NGR-LA	**0.718**	**0.010**	**0.020**	**0.513**
NGR-LA-0P	0.708	0.010	0.019	0.513
NGR-LA-AMB	0.647	0.196	0.301	0.536

Table A34. Comparison of Best F-Score Within the Threshold Span for Each Version of the Classifier for Dependency Parsed Set with NER.

Highest F-score within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.500	0.983	0.662	0.509
PAT-0P	0.499	0.980	0.661	0.508
PAT-AMB	0.490	1.000	0.658	0.490
PAT-LA-0P	0.499	0.982	0.662	0.509
PAT-LA	0.499	0.982	0.662	0.509
PAT-LA-AMB	0.490	0.999	0.657	0.490
NGR-ALL	0.500	0.981	0.662	0.509
NGR-0P	**0.500**	**0.982**	**0.663**	**0.510**
NGR-AMB	0.490	1.000	0.658	0.490
NGR-LA	0.500	0.981	0.662	0.509
NGR-LA-0P	0.500	0.981	0.662	0.509
NGR-LA-AMB	0.490	0.999	0.657	0.490

Table A35. Comparison of Best Accuracy Within the Threshold Span for Each Version of the Classifier for Dependency Parsed Set with NER.

Highest Accuracy within threshold				
	Pr	**Re**	**F1**	**Acc**
PAT-ALL	0.519	0.766	0.618	0.534
PAT-0P	0.519	0.769	0.620	0.535
PAT-AMB	0.611	0.185	0.284	0.531
PAT-LA-0P	0.513	0.805	0.627	0.529
PAT-LA	0.513	0.805	0.627	0.528
PAT-LA-AMB	0.518	0.507	0.513	0.527
NGR-ALL	0.547	0.615	0.579	0.556
NGR-0P	**0.551**	**0.617**	**0.582**	**0.559**
NGR-AMB	0.533	0.512	0.522	0.540
NGR-LA	0.548	0.614	0.579	0.557
NGR-LA-0P	0.549	0.616	0.581	0.558
NGR-LA-AMB	0.536	0.516	0.526	0.543

Table A36. T-Test Results (P-Values) For F-Scores and Accuracies Among All System Results for Dependency Parsed Set with NER.

F-score	PAT-	PAT-	PAT-	PAT-LA	PAT-	NGR-	NGR-	NGR-	NGR-LA	NGR-	NGR-	PAT-
	-ALL	-AMB	-LA	-AMB	-LA-0P	-ALL	-AMB	-LA	-AMB	-LA-0P	-0P	-0P
PAT-ALL		.444	.063	.515	.063	.015*	.056	.021*	.079	.023*	.019*	.019*
PAT-AMB			.000***	.017*	.000***	.672	.005**	.735	.007**	.786	.718	.378
PAT-LA				.008**	.246	.019*	.000***	.018*	.000***	.020*	.021*	.072
PAT-LA-AMB					.007**	.216	.004**	.224	.004**	.239	.230	.557
PAT-LA-0P						.019*	.000***	.018*	.000***	.020*	.021*	.072
NGR-ALL							.353	.093	.458	.014*	.000***	.015*
NGR-AMB								.277	.002**	.251	.323	.048*
NGR-LA									.373	.005**	.506	.019*
NGR-LA-AMB										.340	.422	.067
NGR-LA-0P											.098	.022*
NGR-0P												.018*
PAT-0P												
Accuracy												
PAT-ALL		.045*	.010*	.381	.009**	.001**	.000***	.001***	.000***	.001***	.001***	.065
PAT-AMB			.003**	.042*	.002**	.883	.000***	.955	.000***	.979	.979	.079
PAT-LA				.036*	.356	.001***	.000***	.001***	.000***	.001***	.001***	.008**
PAT-LA-AMB					.034*	.584	.000***	.533	.000***	.495	.496	.484
PAT-LA-0P						.001***	.000***	.001***	.000***	.001***	.001***	.007**
NGR-ALL							.017*	.079	.006**	.004**	.005**	.001***
NGR-AMB								.023*	.002**	.035*	.037*	.000***
NGR-LA									.008**	.019*	.307	.001***
NGR-LA-AMB										.014*	.015*	.000***
NGR-LA-0P											.999	.001***
NGR-0P												.001***
PAT-0P												

$*p \leq 0.05, **p \leq 0.01, ***p \leq 0.001$

About the Authors

Michal Ptaszynski was born in Wroclaw, Poland in 1981. He received a master's degree from the University of Adam Mickiewicz, Poznan, Poland, in 2006, and PhD in information science and technology from Hokkaido University, Japan in 2010. From 2010 to 2012 he was a JSPS postdoctoral research fellow at the High-Tech Research Center, Hokkai-Gakuen University, Japan. Currently, he is an associate professor at the Kitami Institute of Technology. His research interests include natural language processing, affect analysis, sentiment analysis, HCI, and information retrieval. He is a senior member of IEEE, and a member of AAAI, ACL, AAR, ANLP, JSAI, and IPSJ.

Fumito Masui graduated from Okayama University in 1990 with a PhD in Engineering. He worked at Oki Electric Industry Co., was an assistant in the Department of Engineering at Mie University, and a visiting researcher in the Graduate School of Information Science and Technology at Hokkaido University in 2004. Between 2009 and 2018 he was an associate professor in the Department of Engineering at the Kitami Institute of Technology. Since 2018 he has been a professor in the Department of Engineering at Kitami Institute of Technology. His research interests include natural language processing, tourism informatics, and curling informatics. He is a member of the Japanese Society for Artificial Intelligence, the Institute of Electronics Information and Communication Engineers, the Information Processing Society, the Natural Language Processing Society, the Society for Fuzzy Theory and Intelligent Informatics, the Hokkaido Regional Tourism Society, AAAI, ACL, and the Japan Curling Association.

Pawel Lempa was born in Czeladz, Poland in 1987. He received a master's of engineering degree from Cracow University of Technology, Poland, in 2011, and a doctorate of engineering in manufacturing engineering from Kitami Institute of Technology, Japan in 2018. From 2012 to 2015 he was a research and teaching assistant at Cracow University of Technology, and currently holds the same position. His research interests include optimization in construction and natural language processing, mobile programming, and genetic algorithms.

Yasutomo Kimura received his PhD from the Department of Information Science and Technology, Hokkaido University in 2004. He is currently a full professor in the Department of Information and Management Science, Otaru University of Commerce (2005-2007, assistant professor; 2007-2018, associate professor). His research interests include natural language processing, political information processing, and information retrieval. He is a member of the Association for Natural Language Processing, the Information Processing Society of Japan, the Institute of Electronics, Information and Communication Engineers, and the Japanese Society for Artificial Intelligence.

Rafal Rzepka received a master's degree from the University of Adam Mickiewicz, Poznan, Poland in 1999 and a PhD from Hokkaido University, Japan, in 2004. Currently, he is an assistant professor in the Graduate School of Information Science and Technology at Hokkaido University. His research interests include natural language processing, common sense knowledge retrieval, dialogue processing, artificial general intelligence, affect and sentiment analysis, and machine ethics. He is a member of AAAI, JSAI, JCSS, and ANLP.

Kenji Araki received BE, ME, and PhD degrees in electronics engineering from Hokkaido University, Sapporo, Japan, in 1982, 1985, and 1988, respectively. In April 1988, he joined Hokkai-Gakuen University, Sapporo, Japan, where he was a professor. He joined Hokkaido University in 1998 as an associate professor in the Department of Electronics and Information Engineering and became a full professor in 2002. Presently, he is a full professor in the Department of Media and Network Technologies at Hokkaido University. His research interests include natural language processing, spoken dialogue processing, machine translation, and language acquisition. He is a member of the AAAI, IEEE, JSAI, IPSJ, IEICE, and JCSS.

Michal Wroczynski received an MD from the Medical University of Gdansk, Poland in 2001 and trained as a psychiatrist. As a serial entrepreneur, he has worked for 22 years in artificial intelligence systems, founding and managing five technology start-ups in the years 1999-2018, including Fido.ai, Fido Intelligence, Fido Interactive, and Medical Web Design. Currently, he is a CEO and co-founder of Samurai Labs, an AI lab dedicated to eliminating cyberbullying and online violence. He is a co-author and co-leader of five science and research projects financed by the Polish Ministry of Science and Higher Education and is also co-author of Language Decoder technology-based patent applications. Michal has worked as an advisor to government agencies, startups, and NGOs devoted to AI and to combating hate speech and cyberbullying.

Gniewosz Leliwa received a MSc degree from Gdansk University of Technology, Poland, in 2010. From 2010 to 2013, he was a chief linguistic engineer at Fido Intelligence. From 2013 to 2017, he served as the co-founder and a chief science officer of Fido.ai. Since 2010, he has been working as an architect and lead engineer for Fido's core technology—Language Decoder, an automatic knowledge acquisition framework. Currently, he serves as co-founder and director of AI research in Samurai Labs, an AI lab dedicated to eliminating cyberbullying and online violence. His research interests include natural language processing, computational linguistics, information extraction, and knowledge representation and reasoning.

Copyright © 2019 by the Association for Information Systems. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than the Association for Information Systems must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or fee. Request permission to publish from: AIS Administrative Office, P.O. Box 2712 Atlanta, GA, 30301-2712 Attn: Reprints or via email from publications@aisnet.org.