TakeLab

Laboratorij za analizu teksta i inženjerstvo znanja
Text Analysis and Knowledge Engineering Lab
Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva
Unska 3, 10000 Zagreb, Hrvatska

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

BACHELOR THESIS num. 5322

Automated Sarcasm Detection in Social Network Users' Comments

Marin Kukovačec

Zagreb, July 2017.

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING
BACHELOR THESIS COMMITTEE

Zagreb, 3 March 2017

BACHELOR THESIS ASSIGNMENT No. 5322

Student: **Marin Kukovačec (0036485245)**
Study: Computing
Module: Computer Science

Title: **Automated Sarcasm Detection in Social Network Users' Comments**

Description:

Sentiment analysis from social network users' comments has a wide range of applications, including market research, customer experience analysis, political sciences, etc. A major challenge in sentiment analysis from social media texts is the abundant use of sarcasm. Sarcasm, or verbal irony, refers to the statements whose intended meaning is the opposite from the one expressed in text, and which typically serves to mock or convey discontent.

The topic of this thesis is sarcasm detection in telecom users' Facebook comments in Croatian language. Study the methods for sentiment analysis and the methods for sarcasm detection in social network users' comments. Devise and implement a method for sarcasm detection based on machine learning and in-context contrastive sentiment analysis. Compile a suitable dataset for model training and evaluation. Carry out an experimental evaluation of the method on the test set, including a detailed error analysis. All references must be cited, and all source code, documentation, executables, and datasets must be provided with the thesis.

Issue date: 10 March 2017
Submission date: 9 June 2017

Mentor:

Associate Professor Jan Šnajder, PhD

Committee Chair:

Full Professor Siniša Srbljić, PhD

Committee Secretary:

Assistant Professor Tomislav Hrkać, PhD

I would like to thank: Assoc. Prof. dr. sc. Jan Šnajder for mentoring, tutoring, and advising on my thesis, Zoran Medić for providing the data set, and Text Analysis and Knowledge Engineering Lab for providing computing resources.

CONTENTS

1. Introduction

Sarcasm is an ironic or satirical remark that seems to be praising someone or something but is really taunting or cutting (YourDictionary). Example of sarcasm is: "I work 40 hours a week to be this poor." Sarcasm can be used to hurt and offend, or can be used for comic affect as in example: "Don't bother me. I'm living happily ever after." Irony is a message that is intentionally and transparently inconsistent with attitudes or beliefs held between two or more people. In irony there are two message goals, it can represent an opposite result, as in "new tax system will burden those it was intended to help.", but it can also represent a type of speech in which we use words that are the opposite of what you mean, as a way of being funny. Today, social networks are increasingly present in our society and irony is present in many online texts. However, it is difficult to detect it due to the lack of nonverbal communication (Wikipedia), which includes the use of visual cues such as body language (kinesics), distance (proxemics), and physical environments, of voice (paralanguage) and of touch (haptics). So, sarcasm is actually the use of irony to mock or convey contempt.

Area of Computer science that deals with problem of sarcasm detection is natural language processing. This area often relies on machine learning models to solve linguistic problems. Automated sarcasm detection can help service providers know how to react to users comment on their social network page. Training this machine learning model on previous comments with known label can enable it to predict correct label of future comments and detect sarcasm.

Social networks, such as Facebook,[1] were created for the sole purpose of helping individuals communicate. Facebook is largest social network with over one billion daily users. They often express their views, opinions, and feelings sarcastically in posts to receive more likes, mostly from users who understood true meaning of their message. Sarcasm detection in their posts is the task of this thesis. This task is both interesting and challenging natural language problem. Not only that natural language is ambiguous, but irony is also very difficult to understand, especially if topic of post is unfamiliar to person reading it. That can lead to situation that on posts with positive sentiment, but written using words in negative

[1] `www.facebook.com`

sentiment, user replies negatively. Also, an opposite situation occurs more often, when user, out of frustration or anger uses words with positive sentiments, but thinks the opposite. And again, it can occur that some other user, not knowing the background of the post, types positive response. Usually, there is communication of users who are familiar with background of the post, and they will understand true meaning of ironic messages. Most sarcastic messages can be found at Facebook sites of several services, and goods providers. Users express their frustration, and dissatisfaction trough messages, and posts which can often have positive sentiment, and it is service support who should decide how to reply on those. Depending on previous experience with dealing with users posts on providers service, they can be familiar with public opinion of their service. This can help them to determine the meaning of message, and respond accordingly.

In this thesis I will build a machine learning model that will detect sarcasm for Croatian language. Subtasks which will be done are detecting sarcasm on sentence level, detecting sarcasm on sentence level within context, and detecting sarcasm on post level in Croatian language. Given dataset will be split in to train and test dataset uniquely for each subtask. This model will be trained on appropriate dataset containing posts on Croatian Network providers Facebook page which are labeled with "1" if post contain sarcastic message or "0" otherwise. There are several methods which can be used to evaluate this model.

This thesis consists of related work chapter where it is described how other researchers tackled similar problems and what were their results, dataset chapter where it is described what dataset consists of and how is it organized, model chapter where every resource and feature used is described, evaluation chapter describes results achieved and error analysis of the mismatches, and Conclusion at the end to sum everything up.

2. Related Work

Interest for processing, and detecting natural language aspects in social media texts have been rising along with rise of social media. Quite a number of scientific work was published on that topic. Published work on sarcasm, and humor detection, as well as irony detection, in various social medias is relevant to this thesis as they describe used approaches to relevant or similar problem.

Some published work is based on social networks. Reyes et al. (2013) have mined 140,000 tweets from Twitter[1] with #Irony and used four types of conceptual features to detect Irony: Signatures, unexpectedness, style, and emotional scenarios. Signature feature type are concerning pointedness, counter-factuality, and temporal compression. Concerning temporal imbalance, and contextual imbalance fall under feature type: unexpectedness. Style features are character-grams (c-grams), skip-grams (s-grams), and polarity skip-grams (ps-grams). Emotional scenarios include concerning activation, imagery, and pleasantness. They achieved an F1-score of 76 % (baseline was around 68%) with their Decision tree model.

Joshi et al. (2015) harnessed context incongruity to detect sarcasm/irony in tweets. They downloaded tweets with hashtags #sarcasm and #sarcastic as sarcastic tweets, and #notsarcasm and #notsarcastic as non-sarcastic. Their features could be divided into four groups: Lexical, Pragmatic, Implicit congruity, and Explicit incongruity features. They achieved idem score of 61% with SVM classifier which was an improvement of 5 % over baseline.

Riloff et al. (2013) analyzed tweets with #sarcasm. They tried to detect sarcasm by observing sentiment of words in tweet and detecting contrast between positive sentiment, and negative situation. They achieved F1-score of 63% (baseline of 48 %) with SVM classifier.

Reyes et al. (2012) published their work on humor recognition and irony detection on the figurative language of social media. For irony detection they used more abstract features to represent favorable and unfavorable ironic contexts on the basis of profiled polarity, unexpectedness, and emotional scenarios. They achieved F1-score of 79% with Decision tree classifier.

Barbieri i Saggion (2014) have published their work on automatic irony, and humor

[1] www.twitter.com

detection in Twitter. They used dataset prepared by Reyes et al. (2013) which consisted of 40,000 tweets equally divided into four different topics: Irony, Education, Humour, and Politics. They have used seven groups of features: Frequency (gap between rare, and common words), Written-Spoken (written-spoken style uses), Intensity (intensity of adverbs, and adjectives), Structure (length, punctuation, emoticons, links), Sentiments (gap between positive, and negative terms), Synonyms (common vs. rare synonyms use), and Ambiguity (measure of possible ambiguities). They achieved F1-score of 83% for Random Forest classifier.

Published work on irony detection in customer reviews was also done by Reyes i Rosso (2012). They used a collection of customer reviews from the Amazon[2] which were posted by means of an online viral effect. They used n-grams, POS n-grams, funny profiling, positive/negative profiling, affective profiling, and pleasantness profiling features on their model to learn to detect irony. They achieved F1-score of 74,7% (baseline 54 %) using SVM classifier.

Buschmeier et al. (2014) published work on an impact analysis of features in a classification approach to irony detection in product reviews. They have used dataset of 1254 Amazon product reviews, out of which 437 were ironic, and 817 were non-ironic. They achieved F1-score of 74% with their SVM model.

The problem that will be approached in this thesis is similar to the problems mentioned above as their topics fall into natural language processing area. But still, it is different by source of obtaining dataset and one instance of it is post with its attributes, unlike sentences from tweet (Twitter) and review (Amazon). As far as I know, this thesis is first work on sarcasm detection problem for Croatian language.

[2] www.amazon.com

3. Dataset

Dataset consisted of 9676 different posts, comments, and replies to comments from Facebook pages of Croatian network providers. Difference between post, comment, and reply is minor, and it is just their position in post discussion, so they can all be referred to as "post". Every post with property "irony" set to true was considered to be sarcastic. These labels were obtained from prior annotation runs. As seen in Table 3. 1, it was initially provided 9435 posts with 374 posts being marked as sarcastic. I manually collected 241 more posts from Croatian service providers Facebook page, out of which 57 were sarcastic. In total, there are 9676 posts with 431 sarcastic ones.

Dataset	Sarcastic	Non-Sarcastic	Total
Initially provided	374	9061	9435
Manually collected	57	184	241
Total	431	9245	9676

Table 3.1. Overview of dataset

3.1. Dataset Structure

Dataset was provided in JSON format as list of posts. Each post has the following properties:

- "original" which is original text in post without any modification
- "reaction_count" which is number of reactions post received
- "reactions" is list of reactions post received with person who reacted and reaction id
- "sentences" is a list of sentences

– “likes” is list of people names and their ids who liked post
– “likes_count” is number of likes post received
– “emoticons” is list of all emoticons found in post
– “person_id” is id of person who created the post
– “person_name” is name of person who created the post
– “spam” is boolean value saying if post is considered to be spam
– “irony” is boolean value saying if post is ironic, and was also used to tell if post is sarcastic
– “sentiment” is float value describing sentiment of post, negative values representing negative sentiment of post while positive values representing positive sentiment, and 0.0 representing neutral sentiment
– “created_time” is time stamp taken when post was created
– “is_cs” is boolean value telling if post was created by customer support
– “comments”/“replies” is list of comments, and replies (also post) that are response to this post, during normalization they are replaced with “type” property which is 0 if post is top level post, “1” if post is a comment, and “2” if post is a reply.

Post properties:

```
original : String
reaction_count : Integer
reactions : List
sentences : List
spam : Boolean
emoticons : List
person_id : String
likes_count : Integer
irony : Boolean
sentiment : Float
likes : List
person_name : String
created_time : Timestamp
comments : List
is_cs : Boolean
```

Each sentence consists of the following properties: “sentence”, consisting of words, “emoticons”, which is a list of emoticons appearing in sentence, and “sentiment” which is string saying if sentence sentiment was positive, negative or neutral.

Sentence properties:

```
sentence : List
emoticons : List
sentiment : String
```

Each word consist of following properties: “lemma” which is the canonical or citation form of a word, “conll_id” which is position of word in whole post, “pos” is part of speech tag of the word, “dep_tag” is words role in a sentence, and “original” is unmodified form of the word in a sentence.

Word properties:

```
lemma : String
conll_id : Integer
pos : String
dep_tag : String
original : String
```

Class examples:

Example of Post class:

```
original : "Svaka čast koliko ste aljkavi !"
reaction_count : 0
reactions : []
sentences : [
# List of sentences
]
spam : false
emoticons : []
person_id : "1043947978974981"
likes_count : 0
irony : true
sentiment : -1.66667
likes : []
person_name : "Ime Prezime"
comments : [
# List of other replies to this post fragment
]
is_cs : false
type : 0
```

Example of Sentence class:

```
sentence : [
# List of words
]
emoticons : []
sentiment : "positive"
```

Example of Word class:

```
lemma : "svaki"
conll_id : 1
pos : "Pi-fsn-n-a"
dep_tag : "Atr"
original : "Svaka"
```

3.2. Subtask Datasets

The whole dataset was organized in three classes: Post, Sentence, and Word. Each sentence has also reference to Posts that are his followers (they come as comment or reply to this post), and those that are his successors (this post is comment or reply on its successors). In this thesis three distinct, but related subtasks of sarcasm detection will be addressed. For each subtask, unique train and test dataset were selected and every dataset was normalized ("comments"/"replies" property was replaced with type "1" or "2" property).

- **Subtask 1:**

  The First subtask of this thesis is to find sarcasm in isolated sentence fragment from sarcastic post. If a post is sarcastic, the aim of this subtask is to find out whether sentence that gave post its sarcastic sentiment is sarcastic itself. These sentences were selected manually. Sentences were converted into single sentence post keeping some properties from original post. For this subtask, sarcasm detection on sentence level, 115 sentences from sarcastic posts were chosen and put into test dataset. As shown in table 3.2.1, those sarcastic posts were removed from train dataset. Criteria for choosing a sentence among others is different sentiment from other sentences and post. For that purpose sentences were chosen manually from sarcastic posts with more than one sentence to make sure correct sentence is chosen. Sarcasm in those sentences was manually classified. Goal of this test dataset is to see how many of sentences are sarcastic on their own.

	Count
Train dataset	9562
Test dataset	115
Total	9676

Table 3.2.1 Overview of dataset for subtask 1

Since the context of post was erased, the sentence fragment that left had sarcasm label manually reclassified. As shown in table 3.2.2, out of 115 sentences that were

sarcastic on post level with their context, 68 of them were sarcastic by themselves, while 48 sentences were classified as non sarcastic.

Classification	Count
Sarcastic sentences	68
Non sarcastic sentences	47
Total	115

Table 3.2.2 Overview of test dataset for subtask 1

Some examples of sentences from dataset are:

> "A i ti si profesionalan kad ideš na benzi kupovati bon umjesto internet bankarstva u svom pametnom telefonu .. onak zbilja."
> "Da super, ako nisam vip (a nisam) naplacujete cekanje i ne pomognete."
> "Super, znaci ostali ste isti."

- **Subtask 2:**
  The second subtask of this thesis is to find sarcasm in isolated post from its context. The aim of this subtask is to find out if a comment or reply that is sarcastic as part of its surrounding post, comments, and replies will be sarcastic when surrounding post, comment, and replies are removed. Dataset set consist of posts with "type" property "1" (Comments), and posts with "type" property "2" (Replies) converted into posts with type "0" (Top level posts). For this subtask, 100 sarcastic posts with "type" property of greater than 0 were isolated into test dataset, and removed from train dataset, which is shown in table 3.2.3. Sarcasm of each post in test dataset was manually classified. Goal of this dataset is to see if post is sarcastic regardless of his successors, and followers.

	Count
Train dataset	9576
Test dataset	100
Total	9676

Table 3.2.3 Overview of dataset for subtask 2

Reclassification of sentences with their context was required due to all posts, comment, and replies being treated as top level posts. Out of 100 selected sarcastic posts, comments, and replies, 73 of them are sarcastic without outside context, while 27 are not sarcastic, which can be seen in table 3.2.4.

Classification	Count
Sarcastic posts	74
Non sarcastic posts	26
Total	100

Table 3.2.4 Overview of test dataset for subtask 2

Some examples of post senenteces from dataset are:

> "Au kako ste mi spustili, sada mi je život uništen. Niste trebali"
> "Uuuuu hvala na brzim odgovoru .nisam bila bezobrazna niti glasna samo sam 2 put nazvala i molila za pomoc obadva puta dobila istog agenta koji bio prvo ljen mi odgovorit po drugo mi reko da vas nezovem i nepitam gdje nestao novac i jos mi sam reko da mi daje blokadu. Ko je tu bio bezobrazan? Snimate pozive? Ocito ne kad netko takav ko kaze nemojte nas vise zvat jos radi kod vas,da nezadovoljna sam i ako vas nedobijem i mi nitko ne objasni gdje su novci nestale prodjem na drugu mrezu.hvala'
> "Vidim i kultura im je na nivou kad se starijim ljudima obracaju na TI ;)"

- **Subtask 3:**
  The third subtask of this thesis is to find sarcasm in posts with their surrounding post, comments, and replies. For that purpose, out of original dataset, 75 sarcastic, and 75 non sarcastic posts were put together into test_dataset and removed as shown in table 3.2.5. Goal of this test dataset is to see how well model performs on sarcasm detection on balanced dataset.

	Count
Train dataset	9526
Test dataset	150
Total	9676

Table 3.2.5 Overview of dataset for subtask 3

Reclassification of selected posts was not necessary as all posts kept their post level properties and original text, which is shown in table 3.2.6.

Classification	Count
Sarcastic posts	75
Non sarcastic posts	75
Total	150

Table 3.2.6 Overview of test dataset for subtask 3

Some examples of post sentences from dataset are:

"Hvala na isplati odstete :P A ostalima...ako vam izgori Laptop i tv 80.cm ekran preko kabela U VLASNISTVU T-coma dobijete 500 fuckung kuna. Molila bi br.racuna t-coma da Vam ih uplatim pa mi kupite laptop i tv doci cu u Zagreb po njih nije problem!!!! Ugovori za tel,max tv, internet i dva mobitela na pretplatu isticu u 12.mj.a i ovaj 3-ci na bonove cemo zamjeniti! Hvala unaprijed! Vas potrosac 20.godina!"

"Kakva mi korist vi ste ionako uvijek u pravu...lijepi pozz"

"Odlican je ovaj maxtv2go radi tocno 3sekunde"

4. Model

I approached the problem of sarcasm detection as a classification problem. Model was built using supervised machine learning methods. Parameters for model were optimized using grid search.

4.1. Used Resources

Resources are text files containing words that are needed by some features. They were collected manually from the given dataset. Resources are following: amplifier words, common expressions, negative words, positive words, Croatian stop words, and uni grams. Number of words each text file contains is shown in Table 4.1.

Resource	Number of words
Amplifier words	18
Common expressions	22
Negative words	30
Positive words	17
Croatian stop-words	1957
Uni grams	4129

Table 4.1 Overview of resources

- Amplifier words:
  Words that are used in Croatian language to strengthen the meaning of sentence. Some examples are “puno”, “baš”, and “jako”. They are manually collected from posts in dataset.

– Common expressions:
  Expressions that are occur very often in sarcastic posts. They are manually collected from sarcastic posts in dataset. Some examples of those expressions are “svaka vam cast” and “odlicni ste”.

– Negative words:
  Words that give sentences negative sentiment and often appears in posts, comments, and replies on customer service support social network page. Some examples are “raskid”, “račun”, and “problem”. They are manually collected and used to determine sentiment contrast within single sentence.

– Positive words:
  Words that give sentences positive sentiment and often appears in posts, comments, and replies on customer service support social network page. Some examples are all forms of “zadovoljan”, “dobar”, and thanking words like “hvala”. They are manually collected and used to determine sentiment contrast within single sentence.

– Croatian stop words:
  Collection of Croatian stop words. They are used by some features to filter words that are being processed.

– Uni gram words:
  All word lemmas that appear in sarcastic posts along with their weight. Their weight is calculated based on a number of times they appeared.

4.2. Features

The model has 38 features. These features extract information about each post, each sentence of a post, and a post surrounding. They also extract information of how other people responded to post, post sentiment, and post structure. They can be split into four groups which are: Structural features (most used features, used by every model in related work), Semantic features (also used by Barbieri i Saggion (2014), semantic features that are specific to post and sentences are my own idea), Bag of words features (my own idea), and Post level features (my own idea). Distribution of features over groups is shown in Table 4.2.

Feature group	Number of features
Structural features	13
Semantic features	6
Bag of words features	11
Post level features	8
Total	38

Table 4.2 Overview of features

– **Structural features**

Structural features are features that refer to post structure. Those features are usually a float number between 0, and 1 which tells ratio between some property of post, and whole post itself. Features that fall under this category are: Adverb ratio, Attribute ratio, Auxiliary ratio, Commas ratio, Ellipsis ratio, Exclamation mark ratio, Object ratio, Nominative pronoun ratio, Predicate ratio, Preposition ratio, Punctuation ratio, Question mark ratio, and Quotation mark ratio.

- Adverb ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Adv” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Attribute ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Atr” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Auxiliary ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Aux” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Commas ratio:
  A feature which shows ratio between a number of all words in a post having comma as word lemma and a number of all words in a post excluding words tagged with “Punc” word tag.

- Ellipsis ratio:
  A feature which shows ratio between a number of all words in a post having three full stops in a row as original unmodified word and a number of all words in a post excluding words tagged with “Punc” word tag.

- Exclamation mark ratio:
  A feature which shows ratio between a number of all words in a post having exclamation mark as word lemma and a number of all words in a post excluding words tagged with “Punc” word tag.

- Object ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Obj” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Nominative pronoun ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Pnom” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Predicate ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Pred” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Preposition ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Prep” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Punctuation ratio:
  A feature which shows ratio between a number of all words in a post tagged with “Punc” word tag and a number of all words in a post excluding words tagged with “Punc” word tag.

- Question mark ratio:
  A feature which shows ratio between a number of all words in a post having question mark as word lemma and a number of all words in a post excluding words tagged with “Punc” word tag.

- Quotation mark ratio:
  A feature which shows ratio between a number of all words in a post having single or double quotation mark as lemma divided by two (to express number of uses of quotation) and a number of all words in a post excluding words tagged with “Punc” word tag.

– **Semantic features**

  Semantic features are feature group that relies on semantics property of posts and sentences. It uses semantics to determine whether is some post positive or negative, and whether is there contrast inside sentences, posts, comments, and replies. Features that fall into this group are: Negative sentiment feature, Positive sentiment feature, Polarity feature, Polarity with following posts feature, Polarity with succeeding posts feature, and Polarity between sentences inside post feature.

  - Negative sentiment feature:
    A feature which takes sentiment property from post and returns one only if sentiment float value is negative, otherwise returning zero.

- Positive sentiment feature:
  A feature which takes sentiment property from post and returns one only if sentiment float value is greater than zero, otherwise returning zero.

- Polarity feature:
  A feature which returns one if there is sentiment contrast inside one sentence, and zero otherwise. It first loads specific positive, and negative words that are most likely to appear in posts regarding network providing, then goes trough each sentence and takes sentences sentiment. If sentiment of the sentence is positive, it searches for any match of word lemmas with words from negative sentiment words. If sentiment of the sentence is negative, it searches for any match of word lemmas with words from positive sentiment words. If any of those two cases successfully finds a match, then contrast is found and one is returned. Otherwise, zero is returned.

- Polarity with following posts feature:
  A feature which iterates trough posts that come after observed post and count how many of them have sentiment value greater than zero, which are classified as positive, and those with sentiment value lesser than zero, which are classified as negative. Then it checks if both positive sentiment, and negative sentiment counter are greater than zero, and if they are, returns one, or zero otherwise

- Polarity with succeeding posts feature:
  A feature which iterates trough posts that come before observed post and count how many of them have sentiment value greater than zero, which are classified as positive, and those with sentiment value lesser than zero, which are classified as negative. Then it checks if both positive sentiment and negative sentiment counter are greater than zero, and if they are, returns one. Otherwise it returns zero.

- Polarity between sentences inside post:
  A feature which iterates trough posts sentences and count number of sentences with sentiment property being negative, and number of sentences with sentiment property being positive. If both positive, and negative counter are greater than zero then it concludes that there is sentiment contrast between sentences, and returns one. Otherwise it returns zero as there is no sentiment contrast.

– **Bag of words features**

  Bag of words feature group are features that relies on predefined list of words, expressions, symbols, and tries to find them inside sentence and post. They are usually float number between 0, and 1, which is ratio between number of occurrences of observed expression, word or symbol in a post, and maximum number of these occurrences across all posts. Features that fall into this group are: Common expressions feature, Has customer service feature, Has hashtag feature, Has hearth feature, Has laugh feature, Has negation feature, Has negative smiley feature, Has positive smiley feature, Amplifier words feature, Thanking words feature, and Uni gram feature.

  - Common expressions feature:
    A feature which loads common expressions from file containing common expressions collected from sarcastic posts in dataset. It checks for posts original text and counts number of common expression occurrences in it. Return number is ration between common expression occurrences in observed post, and maximum number of common expression occurrences among all posts.

  - Has customer service feature:
    A feature which uses a list of all customer service names and iterate trough each sentence of post to see if any customer service name was mentioned, and returns one if it has appeared, and zero if it has not.

  - Has hashtag feature:
    A feature which uses posts original text to check if it contains hashtag character, and returns one if it does contain it, or zero it is not contained

- Has hearth feature:
  A feature which uses posts original text to check if it contains hearth emoticon "<3", and returns one if it does contain it, or zero it is not contained

- Has laugh feature:
  A feature which uses a list of character sequences that represent laughing in text form, for example: "LOL", "ROFL", "HAHA". It uses post original text, and searches for occurrence of any of character sequence in that original text. If it finds occurrence, it returns one. Otherwise, it returns zero.

- Has negation feature:
  A feature which uses list of negations from Croatian language. It iterates trough posts sentences, and search for any occurrence of negation in any of words in a sentence. If it finds any occurrence of negation in any of words, it will return one. Otherwise, it will return zero.

- Has negative smiley feature:
  A feature which uses list of negative emoticons. It uses posts original text to find any occurrence of negative emoticons in it. It returns one if any occurrence of negative emoticon is found in original text. Otherwise, it will return zero.

- Has positive smiley feature:
  A feature which uses list of positive emoticons. It uses posts original text to find any occurrence of positive emoticons in it. It returns one if any occurrence of positive emoticon is found in original text. Otherwise, it will return zero.

- Amplifier words feature:
  A feature which loads list of words which are used as amplifiers in Croatian language. It iterates trough each of posts sentences and checks if any word lemma is contained within amplifiers list. If contained, a counter is increased. When all words in each sentence is checked, the ratio between counter, and number of all words in whole post without excluding punctuation is returned.

- Thanking words feature:
  A feature which uses list of words which represents thanking in Croatian language. It iterates trough each of posts sentences and checks if any word lemma is contained within thanking words list. If thanking word is found, one is returned. If thanking word is not found in any sentence, zero is returned.

- Uni gram feature:
  A feature which loads words with their weight from file. For each post, It goes trough every sentence in post and for each word, if it is in uni gram list, adds word weight. After that, it returns sum divided by maximum sum.

– **Post level features**

  Post level features is group of features which uses information about post, and post context which can be helpful in detecting sarcasm. Features that fall under this category are: Followers count feature, Successors count feature, Is customer service feature, Spam feature, Post length feature, Likes count feature, Sentence count feature, and Type feature.

  - Followers count feature:
    A feature which shows for each post how many posts, comments, and replies came after it. This feature returns ratio between number of following posts, and total number of post, comments, and replies for one observed top level post.

  - Successors count feature:
    A feature which shows for each post how many posts, comments, and replies came before it. This feature returns ratio between number of succeeding posts, and total number of post, comments, and replies for one observed top level post.

- Is customer service feature:
  A feature which for each post uses its property “is_cs” which tells if creator of post is customer service. If creator of post is customer service, one is returned. Otherwise, zero is returned.

- Spam feature:
  A feature which for each post uses its property “spam” which tells if creator of post is marked as spam. If post is marked as spam, one is returned. Otherwise, zero is returned.

- Post length feature:
  A feature that tells how many words post has. For each post, it iterates trough its sentences, and counts every word that is not punctuation. Feature returns ratio between number of words in post, and maximum number of words in post among all posts.

- Likes count feature:
  A feature which tells how many likes each post received. It takes posts property “likes_count” of post, and for each post returns number of likes that post received.

- Sentence count feature:
  A feature which tells number of sentences in each post. Feature returns ratio between number of sentences in a post, and maximum number of sentences in a post.

- Type feature:
  A feature which tells posts type. Post type “0” is top level post. Post type “1” is comment on post type “0”, and post type “2” is reply to comment (post type “1”). This feature returns post type for each post.

5. Evaluation

Model evaluation was done trough three subtasks: sarcasm detection in a single sentence fragment without context, Sarcasm detection of post without context, and Sarcasm detection of posts with context. Evaluation was done on five different Models across three subtasks. Models that were evaluated were Gaussian Naive Bayes model, Decision Tree classifier, K-Neighbors classifier, Logistic Regression, and Linear Support Vector Classifier. For each subtask, best parameters was found individually for each model using grid search. All above described features were used in every subtask.

5.1. First Subtask: Out-of Context Sentence-level Sarcasm Detection

For the evaluation of the first subtask, three models were trained on train dataset prepared for the first subtask. Training was done on posts with their type, and surrounding context. They were then tested on prepared test dataset and their results were evaluated using the F1-score, as shown in Table 5.1. Logistic regression model with parameter "C" set to 1.74 has scored result of 0.7598. Decision Tree Classifier model has scored F1-score of 0.7093. Linear Support Vector Classifier model with "C" parameter set to 0.1 scored F1-score of 0.7598. Best models were Logistic Regression, and Linear Support Vector Classifier with F1-score of 0.7598.

Model name	F1-score
Logistic Regression	0.7598
Decision Tree Classifier	0.7093
Linear Support Vector Classifier	0.7598

Table 5.1 Results for first subtask

The best models did the same wrong predictions. Errors are observed on Logistic Regression model. Those wrong predictions were mostly made on sentences that are even for

humans difficult to decide whether to classify those sentences without any context as sarcasm or not sarcasm. Some examples of those sentences are:

- *"BRAVO HRVATSKI TELEKOM"* predicted: 1 - actual: 0
- *"Hvala sto postojite"* predicted: 1 - actual: 0
- *"Hvala Vam što imate kvalitetnu podršku!!!!"* predicted: 1 - actual: 0
- *"Bravo Marketing"* predicted: 1 - actual: 0
- *"Evo i mog posta kao još jednog zadovoljnog korisnika"* predicted: 1 - actual: 0
- *"Hvala i veselim se što smo postali tomato korisnici"* predicted: 1 - actual: 0

Other predicting mismatches occur when a sentence contain a word that occurs often in sarcastic posts, or it contains some common expression that occur there. Some examples of those sentences are:

- *"Cestitam!!!"* predicted: 1 - actual: 0
- *"Bravo tele 2!"* predicted: 1 - actual: 0
- *"Ma BRAVO!"* predicted: 1 - actual: 0
- *"Krasno bas."* predicted: 1 - actual: 0
- *"Vrlo profesionalno..."* predicted: 1 - actual: 0

Last group of predicting mismatches are fault of model as it is not perfect and does not work on all cases. Some examples of those sentences are:

- *"Sreća, sreća, radost"* predicted: 1 - actual: 0
- *"Dobar vam taj prasak"* predicted: 1 - actual: 0
- *"Do Bozica ce biti sigurno!!!"* predicted: 1 - actual: 0

5.2. Second Subtask: In-context Sentence-level Sarcasm Detection

For the evaluation of the second subtask, three models were trained on train dataset prepared for the second subtask. Training was done on posts with their type and surrounding

context. They were then tested on prepared test dataset and their results were evaluated using F1-score, as shown in table 5.2. Logistic regression model with parameter "C" set to 7.0 has scored result of 0.8506. Decision Tree Classifier model has scored F1-score of 0.7613. Linear Support Vector Classifier model with "C" parameter set to 0.22 also scored F1-score of 0.8506. Best models were Logistic Regression, and Linear Support Vector Classifier with F1-score of 0.8506.

Model name	F1-score
Logistic Regression	0.8506
Decision Tree Classifier	0.7613
Linear Support Vector Classifier	0.8506

Table 5.2 Results for second subtask

The best models did same wrong predictions. Errors are observed on Logistic Regression model. First group of mismatches occurred on posts that are difficult to classify as they can be both sarcastic and non sarcastic. Some examples are:

- *"Bas sam se nasmijala :P"* predicted: 1 - actual: 0
- *"Bravo plavuso! ;)"* predicted: 1 - actual: 0
- *"Pa Vi ste sjajni!"* predicted: 1 - actual: 0
- *":D baš me veseli da ste ekspeditivni"* predicted: 1 - actual: 0
- *"Zivjeo hrvatski telekom"* predicted: 1 - actual: 0
- *"Molim vas da pokušate nešto napraviti, jer svaki mjesec uvijek problem....prošli put su nešto resetirali u centrali i proradilo je....puno hvala na usluzi...."* predicted: 1 - actual: 0

Second group mismatches occur on post fragments which contained some common expressions found in sarcastic posts. Some examples of those post fragments are:

- *"Oh krasno."* predicted: 1 - actual: 0
- *"Tablet je odlican"* predicted: 1 - actual: 0
- *"Zivio Ime! Zivio tele2!"* predicted: 1 - actual: 0
- *"Vipnet nadam se da pratite...zadovoljni su ljudi... jako zadovolnji....."* predicted: 1 - actual: 0

Last group of mismatches are mismatches made by fault of model. Some examples of those mismatches are:

- *"Ahahahahahahhahaha :)"* predicted: 1 - actual: 0
- *"Jeftin vam je Alcatel 20.45 One Touch Prezime ime"* predicted: 1 - actual: 0
- *"20.45,Ima i poklopac, 256 k boja, uz surfam ga dobijete bez naknade i otplate na rate, a giga i pol neta je i vise nego dovoljno na njemu"* predicted: 1 - actual: 0

5.3. Third Subtask: Post-level Sarcasm detection

For the evaluation of the third subtask, five models were trained on train dataset prepared for the third subtask. Training was done on posts with their type, and surrounding context. They were then tested on prepared test dataset and their results were evaluated using accuracy score, as shown in table 5.3. Logistic regression model with parameter "C" set to 2.3 has scored result of 0.68. Gaussian Naive Bayes model scored accuracy score of 0.7867. Decision Tree Classifier model with "criterion" parameter set to "gini", "random state" parameter set to 100, "max depth" parameter set to 5, and "min samples leaf" parameter set to 5 has scored accuracy score of 0.6267. K neighbors classifier has scored accuracy score of 0.5267. Linear Support Vector Classifier model with "C" parameter set to 0.735 scored accuracy score of 0.6867. Best model was Gaussian Naive Bayes with accuracy score of 0.7867.

Model name	Accuracy Score
Logistic Regression	0.68
Gaussian Naive Bayes	0.786667
Decision Tree Classifier	0.6267
K Neighbors Classifier	0.5267
Linear Support Vector Classifier	0.686667

Table 5.3 Results for third subtask

Mismatches on the third subtask are observed on best model, which is Gaussian Naive Bayes. The first group of mismatches that occurred on third subtask when model predicted

their label to be “0” (Non sarcastic), while they were labeled as sarcastic. In some of those predictions, it was difficult even for human to know whether post is sarcastic or not. Some examples of posts are:

- *“Bitno je da ne priznate grešku, čini se ;)”* predicted: 0 - actual: 1
- *“Lijepo od vas Ime..”* predicted: 0 - actual: 1
- *“Iznimno korisna informacija kada nemas signal (Y)”* predicted: 0 - actual: 1
- *“Da Arena i Eurosport je kao RTL i HBO znate?”* predicted: 0 - actual: 1

Other wrong predictions that fall under first group are due to model faulty. Some of examples that fall under this category are:

- *“klap klap”* predicted: 0 - actual: 1
- *“Divno, ma blago nama s vama :)”* predicted: 0 - actual: 1
- *“Ime,Ime je u akciji-zamijenio Ime...*
  imaj povjerenja u njega....Ime-svi smo
  uz tebe-nedaj se!!!!! :D :D :D” predicted: 0 - actual: 1

The second group of incorrect predictions are predictions that post is sarcastic, when the actual label is non sarcastic. In some cases, it is difficult to determine whether post is actually sarcastic or not, like in examples:

- *“Zbilja ste azurni.... opet cekam odgovor od sluzbe ...*
  a da ne kazem da sam u utorak 30 min cekal na otvorenoj
  vezi sa operaterom i nista nisam rijesio. Al nema veze
  doci ce 11mj. pa cu vas lijepo pozdravit i otici u drugu
  mrezu. Jer ovo je vec previse kak se lose odnosite prema
  potrosacima i korisnicima.da ne kazem da opomenu
  bez pardona saljete al nebi se potrudili sto prije rijesiti
  upit korisnika.” predicted: 1 - actual: 0
- *“A mene zanima dali mogu ja dobiti poklon poslije 7*
  godina pretplate i visoke tarife,Hrvatski Telekom,” predicted: 1 - actual: 0

Other mismatches are due to model imperfection. Those posts have some common expressions that are used in sarcastic posts, but they are not sarcastic. Examples of those posts are:

- *“Znam, želite li s drugog mobitela i iste lokacije*
  brzinu 4g od konkurencije ili bolje da ne stavljam ;)” predicted: 1 - actual: 0

– *“Nakon cca 3 mj odustala sam od dopisivanja jer njihovi odgovori nisu imali veze sa pameću i stvarnom situacijom, nisu nikada dali konkretan odgovor a telefonski NE MOŽEŠ dalje od službe za korisnike.”* predicted: 1 - actual: 0

6. Conclusion

Sarcasm in social networks is most of the time determined by examining sentiment of comment with sentiment of surrounding post, comments, and replies. It can also be detected by finding positive sentiment words, and negative sentiment words inside same sentence. It is common that a user, who is frustrated with network service provider, writes a series of sentences with negative sentiments, and throws in one positive where he sarcastically praises the network provider for their service. Also, situation that often occurs is a sarcastic reply in one sentence to network providers customer services comment. In such a situation, it is necessary to observe context of whole discussion from top level post to this specific reply. The solution described in this thesis had some success in detecting sarcasm in all three subtasks. However, sarcasm can sometimes be difficult to detect even for humans. There are numerous examples of posts that can be classified as sarcasm, and not sarcasm at the same time. This obstacle is difficult to overcome by machines. Problem of sarcasm detection falls under natural language processing problems in computing. With machine learning model I built, I achieved F1-score of 0.7598 in out-of context sentence-level sarcasm detection subtask, F1-score of 0.8506 in in-context sentence-level sarcasm detection subtask, and accuracy score of 0.7867 in post-level sarcasm detection subtask.

In future work on sarcasm detection, it would be good to collect more examples of sarcastic posts in order to reduce huge dataset imbalance. It would also be good to think of some other features other than 38 described in this thesis which would contribute to sarcasm detection in comments. A bit more difficut subtask which can be done in field of sarcasm detection is likehood (in percentage) of sarcasm in some sentence or post.

BIBLIOGRAPHY

Francesco Barbieri i Horacio Saggion. Automatic detection of irony and humour in twitter. U *Proceedings of the International Conference on Computational Creativity*, 2014.

Konstantin Buschmeier, Philipp Cimiano, i Roman Klinger. An impact analysis of features in a classification approach to irony detection in product reviews. U *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, stranice 42–49, 2014.

Aditya Joshi, Vinita Sharma, i Pushpak Bhattacharyya. Harnessing context incongruity for sarcasm detection. U *ACL (2)*, stranice 757–762, 2015.

Antonio Reyes i Paolo Rosso. Making objective decisions from subjective data: Detecting irony in customer reviews. *Decision Support Systems*, 53(4):754–760, 2012.

Antonio Reyes, Paolo Rosso, i Davide Buscaldi. From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Engineering*, 74:1–12, 2012.

Antonio Reyes, Paolo Rosso, i Tony Veale. A multidimensional approach for detecting irony in twitter. *Language resources and evaluation*, 47(1):239–268, 2013.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, i Ruihong Huang. Sarcasm as contrast between a positive sentiment and negative situation. U *EMNLP*, svezak 13, stranice 704–714, 2013.

Wikipedia. Nonverbal communication. `https://en.wikipedia.org/wiki/Nonverbal_communication`. Accessed: 2017-04-21.

YourDictionary. Examples of sarcasm. `http://examples.yourdictionary.com/examples-of-sarcasm.html`. Accessed: 2017-04-21.

Automated Sarcasm Detection in Social Network Users' Comments

Abstract

Sarcasm is an ironic or satirical remark that seems to be praising someone or something but is really taunting or cutting. Field of computer that deals with problems like sarcasm detection is natural language processing, which often uses machine learning to yield best results. This Bachelor thesis describes solution implementation of automated sarcasm detection in users comments on Facebook social network. This task of sarcasm detection can be split in three subtasks: Sarcasm detection in sentence fragments taken from sarcastic posts, Sarcasm detection in post fragments without their outside context, and Sarcasm detection in posts with their outside context. Each subtask required unique splitting of original dataset into train dataset to train the model, and test dataset to evaluate the model. This thesis describes dataset of each subtask, solution implementation, and evaluation results on test datasets for each subtask.

Keywords: Natural language processing, Machine learning, Sarcasm detection, Social networks, Croatian Language

Strojno otkrivanje sarkazma u komentarima korisnika društvenih mreža

Sažetak

Sarkazam je ironična ili satirična primjedba koja izgleda kao da nekoga hvali, ali zapravo ga ismijava. Područje računarske znanosti koje se bavi problemima poput detekcije sarkazma je obrada prirodnog jezika, koja često koristi strojno učenje kako bi polučila najbolje rezultate. U ovom završnom radu opisana je implementacija rješenja za problem detekcije sarkazma u komentarima korisnika na društvenoj mreži Facebook. Unutar problema koji se obrađuje nalaze se tri podzadatka: detekcija sarkazma u rečenicama izvađenim iz sarkastičnih komentara, detekcija sarkazma u postovima bez vanjskog konteksta, te detekcija sarkazma u postovima sa kontekstom. Svaki podzadatak je zahtijevao poseban skup podataka za treniranje modela, kao i poseban i jedinstven testni skup podataka za evaluaciju modela. U ovom radu su opisani izgled skupa podataka svakog podzadatka, sama implementacija rješenja i rezultati evaluacije nad testnim skupom podataka za svaki od podzadataka.

Ključne riječi: Obrada prirodnog jezika, Strojno učenje, Detekcija sarkazma, Društvene mreže, Hrvatski jezik