#HashtagWars: Learning a sense of humor (SemEval 2017)

Miha Pešič, Rok Zidarn

January 14, 2017

Abstract

Humor je eden izmed poglavitnih karakteristik človekovega obstoja in je morda celo nek pokazatelj inteligence. Čeprav ljudje povsem enostavno klasificiramo humor, je na področju računalništva in avtomatske detekcije ter klasifikacije humorja še vedno težava, kako oceniti humor. Trenutno je že precej težavno določiti binarni razred, bodisi je nekaj smešno bodisi ne, kaj šele podati oceno stopnje humorja, npr 10 - zelo smešno, 1 - povsem ne smešno. Težava izhaja predvsem iz tega saj je potrebno veliko predznanja, humor je tudi precej subjektiven, lahko se pojavi sarkazem ali ironija. Namen te naloge bi torej bil razviti nek klasifikator, ki bi čim bolje napovedal ali bo nekaj smešno ali ne. Izhajali bomo iz podatkov oddaje midnight, kjer imajo med humoristi tekmovanje imenovano #hashtagwars v katerem tekmujejo, kdo bo napisal čimbolj smešen tvit (ang. tweet) na določeno temo (ang. hashtag).

1 Introduction

Poročilo je razdeljeno na več poglavij. V prvem bomo predstavili uvod v problem s, katerim se bomo ukvarjali in malce osnov, ki se tičejo humorja in njegovih vrst. Nato bodo predstavljene metode s, katerimi smo se lotili reševanja problema, natančneje bo predstavljena tudi oblika podatkov, postopek predprocesiranja in ideja rešitve. V tretjem poglavju bomo predstavili rezultate, ki smo jih dobili in kaj ti rezultati sploh pomenijo. V zadnjem poglavju pa bomo primerjali našo rešitev z obstoječimi, predstavili bomo možne izboljšave in diskutirali rezultate.

Na področju umetne inteligence in procesiranja naravnega jezika je napovedovanje in ocenjevanje humorja v tekstu ena izmed bolj zanimivih nalog v zadnjem času. Ljudem je ponavadi povsem jasno, kdaj gre za šalo in kdaj ne, vendar sistemom kot so računalniki pa ne povsem. Potrebno je neko predznanje o sami temi šale, humor se lahko stopnjuje, nadaljuje, pojavi se lahko sarkazem, ironija. Včasih pride do pretiravanja ali igre besed (ang. puns). Kljub temu, da že obstajajo neki algoritmi in klasifikatorji trenutno lahko napovedujejo zgolj binarni razred, ali je tekst smešen (da/ne). Zaželjeno bi torej bilo, da bi lahko ocenili stopnjo humorja (1-10), vendar tukaj se bomo osredotočili na napovedovanje binarnega razreda, smešno (da/ne). Uporabili bomo podatkovno zbirko tvitov, spisanih v oddaji midnight, kjer so podatki o temi (hashtag-u), tekstu tvita in oceni. Ocena je določena z eno izmed treh vrednosti, 0 pomeni, da tvit ni smešen, 1 pomeni, da je tvit smešen, 2 pa, da je tvit najbolj smešen v podani temi. Seveda ni nujno, da bomo iz teh podatkov zgradili dovolj dober klasifikator, kajti to oceno je podala zgolj majhna skupina ljudi in potrebno je zopet poudariti, da je humor subjektivna zadeva.

V prvem pog

1.1 Humor

Ljudje smo družabna bitja, za obstoj potrebujemo druge in humor je eden izmed načinov povezovanja, komunikacije, kakor tudi način sproščanja. Humor je tesno povezan s smehom oziroma z dobrim počutjem, kajti med tem procesom se sproščajo endorfini, ki jim drugače rečemo hormon sreče. Glede na to, da je humor subjektiven imamo ljudje različne okuse humorja, nekaterim je nekaj smešno, drugim ne. Kar vpliva na to je morda socialni status, vzgoja, družba ali kultura, zaradi česar ne moremo humorja enostavno opisati oziroma formalno zapisati.

1.2 Vrste pisanega humorja

Poznamo več vrst zapisanega humorja, oziroma več značilk, ki opisujejo vrsto humorja. Spodaj je zapisanih nekaj primerov:

1. Fonologija: “What do you use to talk to an elephant? An elly-phone."
2. Nasprotja: “Mythical Institute of Theology" - (MIT)
3. Negativna orientacija: “Money can’t buy your friends, but you do get a better class of enemy."
4. Stereotipi: “It was so cold last winter that I saw a lawyer with his hands in his own pockets."
5. Aliteracija: “Infants don’t enjoy infancy like adults do adultery."
6. Antonomija: “Always try to be modest and be proud of it!"

2 Methodology

Glavna metoda s katero smo se ukvarjali je bila klasifikacija. Namen je klasificirati oziroma določiti ali je neka vsebina smešna ali ne. Torej napovedovali smo binarni razred (smešno/ne) na podlagi podatkov, ki so zapisani v obliki tvitov. Problem smo poskušali rešiti tudi z uvrščanjem (angl. clustering), drugače pa je bil cilj ugotoviti, katere značilke uporabiti pri klasifikaciji, oziroma, katere značilke vplivajo na humor v tvitu. Za to smo uporabili t.i. funkcije značilk (angl. feature function), na podlagi, katerih se je potem naš model učil. Boljpe funkcije značilk bomo uporabili, bolje se bo naš model naučil napovedovanja, boljši bodo rezultati, ki jih bomo predstavili v poglavju Rezultati.

2.1 Podatki

Za reševanje problema klasifikacije humorja so nam bili na voljo podatki iz oddaje midnight, kjer so si zbrani gostje, komiki, morali izmisliti tvit, kjer bi uporabili določen hashtag, ki predstavlja neko temo. Na primer na temo, ki je določena z hashtag-om #DogBooks, so si morali izmisliti, čim bolj smešne tvite v povezavi s knjigami o psih. Bodisi so to naslovi knjig ali besedne igre. Na podlagi svojih predlogov tvitov so bili tudi ocenjeni, ali je bil njihov predlog smešen ali ne.

Ti podatki so zbrani v 2 direktorijih. Enega smo uporabili za učenje našega modela klasifikatorja, drugega pa za napovedovanje in merjenje uspešnosti naše rešitve. Znotraj tih dveh direktorijev je na voljo več datotek, katerih se vsaka nanaša na določeno temo oziroma hashtag, kjer so zbrani tviti s to temo.

2.2 Predprocesiranje

Podatki, ki so zapisani znotraj datotek, ki imajo končnico .tsv, so v tviti, ki so bili objavljeni na Twitter-ju in dodatno označeni s hashtag-om določene teme. Primer ene vrstice v teh datoteka izgleda tako:

712653798003621888 @midnight Bone With the Wind *#DogBooks 0*

Najprej je zapisan ID tvita, ki je predstavljen v obliki števila, nato je tekst tvita, ki je bil objavljen in na koncu ocena. Ocena 0 predstavlja, da tvit ni bil ocenjen kot smešen, 1 pomeni, da je tvit smešen in ocena 2 pomeni, da je tvit najbolj smešen znotraj te teme oziroma hashtag-a.

Potrebno je tudi poudariti, da je pri samem tekstu v tvitu bilo ponekod, kar nekaj odvečnih podatkov, ki smo jih morali pred samo klasifikacijo odstraniti, saj bi drugače vplivali na rezultate. Sem spadajo dodatni hashtag-i poleg osnovnega, ki se nanaša na temo, razne hiperpovezave in oznaka midnight.

Podatke o tvitu smo nato shranili v objekt oziroma razred, ki ima atribute id, hashtag, text in tokens. Atribut tokens je pomemben atribut, kajti večino funkcij značilk bo delovalo v povezavi z njim. Namreč, sem smo shranili posamezne besede, ki se pojavi v tekstu tvita, ki smo lematizirane, odstranjeni so dodatni presledki in stop besede, kot so vezniki, mašila in podobno.

Za pridobivanje čim kvalitetnejših značilk smo uporabili tudi dodatne podatke, kot zbirke slengovskih, pozivitivnih, negativnih besed in emoticon-ov.

2.3 Značilke

Pri klasifikaciji smo uporabili večje število značilk, ki smo jih pridobili s pomočjo funkcij. Ta so bodisi vračala binarno vrednost true/false, razmerja določenih lasnosti teksta, razdalje bodisi razne mere. Spodaj so naštete in opisane uporabljene funkcije:

- ratioOfCapitalLettersFF(text)

  Izračuna razmerje med številom besed, ki se začnejo z veliko začetnico in vsemi besedami oziroma dolžino teksta

- ratioOfStopWordsFF(text)

  Izračuna razmerje med številom stop besed in dolžino teksta

- containsPunctuationFF(text)

  Vrne logično vrednost (true/false), bodisi tekst vsebuje ločila ali jih morda ne, preverja pike, vejice, podpičja, vprašaje in podobno

- ratioOfLemmasFF(tokens, text)

  Izračuna razmerje med vsemi besedami, vključno z stop besedami, emoticoni in podobnim, ter lematiziranimi besedami, če jih ne najde vrne vrednost 0

- containsMostCommonWordFF(tokens, mostCommonWord)

  Najprej se iz učne množice ugotovi, katera je najpogostejša beseda, pri tem so izločene stop besede, nato v tekstu pregleda ali se ta beseda nahaja notri, če se vrne logično vrednost true

- cosineSimilarityToPunFF(text, pun):

  Na podlagi teme oziroma hashtag-a na spletnem viru preišče bazo besednih iger s podobno vsebino. Pridobi te podatke in nato izračuna kosinusno podobnost med tekstom besedne igre in tekstom tvita

- getSentimentScoresFF(text):

  S pomočjo modula SentimentAnalyzer izračuna mero pozitivnega in negativnega sentimenta v tekstu

- containsProfanityFF(text, profanityWords):

  Vrne logično vrednost ali tekst vsebuje žaljive besede ali ne

- containsNegativeWordsFF(text, negativeWords):

  Vrne logično vrednost ali tekst vsebuje negativne besede ali ne, na podlagi zbirke podatkov

- positiveToNegativeWordRatioFF(text, negativeWords, positiveWords):

  Vrne razmerje med pozitivnimi in negativnimi besedami, v primeru, da ne najde podatkov vrne vrednost 1

- calculateVerbToNounRatioFF(tokens)

  Izračuna razmerje med številom glagolov in samostalnikov v tekstu, če se ne pojavita ti skupini besed, vrne vrednost 0

- containsEmoticonsFF(text, emoticonList)

  Vrne logično vrednost ali besedilo vsebuje emoticone ali ne

- containsSlangFF(text, slangList)

  Vrne logično vrednost ali besedilo vsebuje sleng ali ne

- calculateLexicalDiversityFF(text)

  Izračuna leksikalno raznolikost, izogibanje ponavljanju besede

- calculateSemanticRelatednessFF(tokens)

  Izračuna maksimalno semantično sorodnost, na podlagi para besed, pridobi njune synsete, izračuna podobnost ter vrne maksimalno vrednost nekega para

- hypernymRepetitionFF(tokens)

  Vsako besedo primerja z ostalimi iz teksta, pridobi njuna hipernima (nadpomenki) in v primeru, da imata ti besedi isto nadpomenko vrne logično vrednost true, drugače false

Nekatere izmed funkcij značilk so precej osnovne in nekako niso preveč v povezavi s samo detekcijo humorja, vendar smo jih kljub temu uporabili, zaradi preprostosti in radovednosti, ali so morda sploh uporabne. Sem spadajo na primer razmerja besed, ki se začnejo z veliko začetnico, razmerje lem, ali tvit vsebuje najbolj pogosto besedo in podobne.

Nekatere so malce bolj uporabne, vendar so bile kljub temu enostavno implementirane, na primer ali tvit vsebuje žaljive besede, kar je v današnjem svetu pogost pojav, saj se velikokrat tako le še poudari absurd ali ironija v šali. Za nekatere pa predvidevamo, da nam bodo dala precej globjo informacijo o humorju, ki ga morda vsebuje tvit, kot je na primer razmerje med negativnimi in pozitivnimi besedami, kajti za nekatere ša le je značilno bodisi so sarkastične, kjer se pojavljajo predvsem pozitivne besede ali negativne besede pri šalah o stereotipih.

Ker se pojavljajo v šalah tudi besedne igre, preverjamo ali obstajajo takšni zapisi že v neki bazi s temi podatki, tukaj računamo predvsem podobnost tvita, na podlagi TDIDF, z besednimi igrami s podobno vsebino in če sta si teksta podobno to morda nakazuje na šalo. Za izražanje dvoumnosti smo tudi uporabili postopek POS označevanja, saj lahko neko besedo morda uporabimo kot glagol, v šali pa pride do izraza kot samostalnik.

Absurd je prav tako sestavni del šal, saj predstavlja neko hitro spremembo stanja, nerealno situacijo in temu je namenjena primerjava nadpomenk. Torej če je razdalja med synset-oma nadpomenk velika pomeni neko drastično spremembo v tekstu in s tem humor. Ironija se morda pojavlja v tvitih z uporabo emoticon-ov, ki predstavljajo čustva, nekdo lahko zapišemo neko precej negativno zadevo zraven pa izrazi pozitivno čustvo. Z izračunom semantične sorodnosti pa ciljamo predvsem na to, da humor vpliva tudi nepričakovanost, torej če ima nek tvit nizko semantično sorodnost to morda pomeni humor.

2.4 Klasifikacija

Na podlagi spisanih funkcij značilk bo delovala naša klasifikacija. Klasifikator oziroma model se bo na podlagi lastnosti, ki bodo definirana z značilkami ter predhodno definirano vrednostjo humorja tvita naučil, kako so zgrajeni smešni tviti in kako ne, ter bo napovedoval na testni množici podatkov. Pri tem bomo uporabili klasifikacijsko metodo imenovano Naivni Bayes.

Naivni Bayes je enostaven klasifikator, ki temelji na verjetnosti kateremu razredu pripada trenutni element, poleg tega predvideva neodvisnost dogodkov. Ker izračunavamo večje število značilk jih lahko uporabimo kot nek vektor, zato uporabljamo Multinomial Naive Bayes, ki nam nato vrne vektor verjetnosti, kjer neko število predstavlja, kolikšna je verjetnost, da ta element pripada določenemu razredu.

Najprej se bomo na večji količini podatkov, ki so v direktoriju train_data naučili oziroma bo to namesto nas storil klasifikator na podlagi značilk in njihovih vrednosti. Nato pa bomo napovedovali razred na podatkih iz direktorija trial_data, katere nismo uporabili za učenje, vendar imajo kljub temu zraven zapisane ocene tvitov, na podlagi, katerih bomo nato ocenili uspešnost napovedovanja.

3 Results

4 Discussion