Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 2, 842 16 Bratislava 4

Adam Mikolašek, Veronika Žatková

Určovanie sentimentu filmových recenzií

Projekt z predmetu NSiete

Študijný program: Inteligentné softvérové systémy
Ak. rok: 2019/2020, zimný semester
Cvičiaci: Ing. Matúš Pikuliak

Klasifikácia sentimentu vo filmových recenziách

1. Motivácia

Veľké množstvo ľudí sa pri výbere filmov riadi práve názormi a recenziami iných. Určenie sentimentu sa neobmedzuje iba na recenzie, ale je využiteľné pri drvivej väčšine rečových prejavov.

Schopnosť určiť sentiment recenzií nemusí byť len finálnym produktom, ale aj základom pre ďalšie analýzy. Pokiaľ vieme určiť sentiment recenzií filmu, agregáciou sentimentov identifikovaných pri filme vieme odhadnúť celkový názor verejnosti na daný film a využívať ho ako alternatívny zdroj hodnotenia filmov. Určenie negatívneho sentimentu je zároveň jedným z prvých krokov pre pomoc pri identifikovaní úmyselne nenávistných komentárov.

2. Súvisiace práce

Dataset, ktorý sme pre danú problematiku našli, bol už využitý v inej práci[1]. Cieľom tejto práce bolo určovanie sentimentu a sémantiky textu, a využitie týchto informácií na klasifikáciu komentárov do dvoch skupín a to podľa toho či komentár patril k pozitívnemu alebo negatívnemu hodnoteniu. Výsledky navrhnutého modelu boli testované na dvoch datasetoch, “Pang and Lee Movie Review Dataset” a “Large Movie Review Dataset”, pričom druhý z týchto datasetov je bližšie opísaný v kapitole 3. Výsledky, ktoré navrhnutý model dosiahol boli porovnané s rôznymi inými technikami na spracovanie prirodzeného jazyka, ako sú Bag-of-words, Latent semantic analysis (LSA) a Latent Drichlet allocation (LDA). Navrhnutý model dosiahol najlepšie výsledky práve v spojení s technikou Bag-of-Words a to 88.89, teda v 88.89% prípadoch kategorizoval komentár správne.

[1] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. (2011). Learning Word Vectors for Sentiment Analysis. *The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011).*

3. Dataset

V rámci nášho projektu sme sa rozhodli využiť dataset filmových recenzií “Large Movie Review Dataset v1.0”[2], publikovaný univerzitou Stanford. Dataset pozostáva z 50 000 výrazne polárnych filmových recenzií, mapovateľných na URL adresu filmu, ku ktorému patria (URL sú súčasťou datasetu). Recenzie sú oanotované - každá recenzia má priradenú hodnotu “pozitívna” alebo “negatívna” na základe jej sentimentu. Celková početnosť týchto dvoch tried je vyvážená - 25 000 rezencií je pozitývnych a 25 000 negatívnych. Výrazná polarita recenzií znamená, že negatívne recenzie patria k hodnoteniam <= 4 z 10 a pozitívne recenzie k hodnoteniam >= 7 z 10.

Maximálny počet recenzií pre jeden film je 30 (pre predídenie predpovedania rovnakých hodnôt sentimentu pre jeden film). Dataset je taktiež rozdelený na trénovaciu a testovaciu množinu v rovnakom pomere (25 000 : 25 000), pričom platí, že žiaden z filmov v trénovacej množine sa nenachádza v testovacej množine. K recenziám je taktiež dostupné ich pôvodné hodnotenie vo formáte počtu bodov (maximum je 10).

Súčasťou datasetu je aj ďalších 50 000 recenzií filmov, ktoré sú však neanotované (napríklad pre využitie v rámci metód učenia bez učiteľa). Jedná sa najmä o neutrálne recenzie.

4. Návrh riešenia

Naše riešenie pozostáva z viacerých krokov:

1. Exploratívna analýza dát.
2. Predspracovanie dát - v našom prípade výrazne využívajúce prístupy spracovania prirodzeného jazyka
3. Vytvorenie reprezentácie dát vhodné pre vstup do neurónovej siete (embedding) vrátane samotného výberu vhodného embeddingu.
4. Návrh architektúry a implementácia neurónovej siete, trénovanie a validácia.
5. Vyhodnotenie výsledkov.

[2] http://ai.stanford.edu/~amaas/data/sentiment/

5. Implementácia

Zadanie sme implementovali pomocou jazyka Python, v prostredí Jupyter notebook-ov. Pre prácu s dátami využívame najmä knižnice Pandas a Numpy.
Pre spracovanie jazyka sme využili kniznicu NLTK. Pre implementovanie neurónových sietí používame Keras.

5.1 Predspracovanie dát

Prvým krokom pre predspracovanie dát je načítanie datasetu. Načítavané dáta boli opísané v kapitole 3. Každý načítaný záznam obsahuje tieto informácie:

- Comment - Text hodnotenia
- Sentiment - Hodnota 0 alebo 1 podľa toho či ide o pozitívny alebo negatívny komentár

Druhým krokom je spracovanie textu komentárov, teda NLP (spracovanie prirodzeného jazyka). Spracovanie jazyka začína upravením textu (odstránením skrátených foriem slov, viacerých medzier zasebou a podobne). V prvej iterácii tohto projektu sme zadefinovali stopwords - slová, ktoré textu neprinášajú význam - a odstránili ich. V ďalších iteráciách sme sa ich však rozhodli ponechať, nakoľko v kontexte skladby celej vety môžu mať pre neurónku stále nejakú výpovednú hodnotu.
.Poslednou časťou spracovania komentárov je prevedenie slov do základnej formy. Na toto sme zatiaľ použili stemmer (Porter stemmer z knizžnice nltk).

5.2 Embedding

Embedding začína vytvorením slovníka (dictionary). Slovník je vytvorený zo slov nachádzajúcich sa v trénovacej časti datasetu a je zoradený podľa počtu výskytov slov. Toto je z dôvodu ak by sme nechceli používať celý slovník ale iba istú časť najčastejšie sa vyskytujúcich slov.

Slovník má veľkost 79709 slov, pričom na indexe 15000 je slovo vyskytujúce sa 9x, a na indexe 35000 už len slovo vyskytujúce sa 2x. Zatiaľ sme obmedzili slovník na 15000 najčastejších slov.

Keďže slovník je vytvorený z trénovacej množiny, v testovacej množine sa môžu nachádzať slová, ktoré v slovníku nie sú - tieto slová sú odstránené.

Dáta vstupujúce do neurónovej siete musia mať rovnaké dĺžky. Túto dĺžku sme nastavili na 100 slov (toto číslo sa ešte môže zmeniť), a dlhšie komentáre sme orezali. Kratšie komentáre sme doplnili nulami - zero padding.

Pri embeddingu máme možnosť natrénovať vlastný (čo má výhodu silného zastúpenia doménových slov) alebo použiť už vytvorený. V ďalšej iterácii projektu sme vytvorili vlastný embedding pomocou knižnice Fasttext, ktorý pre každé slovo v našej slovnej zásobe (15000 slov) vytvoril vektor hodnôt ako jeho embedding. Embeddingy sme následne použili ako vstupy do Embedding vrstvy v neurónke. Embedding pre každé slovo mal veľkosť 100.

5.3 Neurónová sieť

Architektúra siete je ‘many_to_one’, keďže na vstupe máme väčší počet slov a výstupom je 1 label. Jednotlivé vrstvy sú sekvenčne zoradené za sebou:

Obr. 1 - architektúra základného modelu NN.

Použité vrstvy v neurónovej sieti:

- Embedding vrstva
  - Input_dim o veľkosti vocab-u - v našom prípade 150001
  - Ouput_dim o veľkosti embedding-u - 100
  - Maska nastavená na nuly (kvôli zero paddingu)
- Bidirectional vrstva
  - Aj vstupný aj výstupný layer je lstm s veľkosťou 64
- Dense vrstva
  - Output vrstva
  - Aktivačná funkcia sigmoid
  - Vystupuje z nej 1 hodnota
- Dropout vrstva
  - Pridaná kvôli pretrénovaniu modelu (overfitting)

Tieto parametre a vrstvy môžu byť ešte v ďalšej časti projektu upravené alebo zmenené.

6. Vyhodnotenie

Pri vyhodnocovaní správnosti klasifikácie sentimentu, ktorý je problémom **binárnej klasifikácie**, sme si ako metriku zvolili **presnosť** (accuracy) - nakoľko máme vyvážený dataset, táto metrika je vhodná. Loss funkciou je **binary cross-entropy** (logaritmická chyba).

Každý model sme trénovali na datasete 25000 filmových recenzií, z čoho polovica bola pozitívna a polovica negatívna. Testovací dataset bol rovnakej veľkosti a rovnakého zastúpenia tried, takže naše trénovacie a testovacie dáta boli v pomere 50:50 (toto môže byť atypické, nakoľko väčšinou sa tento pomer pohybuje okolo hodnôt 75:25 alebo 80:20, avšak nakoľko bola trénovacia množina dosť veľká, dataset sme nechali v pôvodnom rozdelení od jeho autorov).

Náš prvý baseline model dosiahol po 10 epochách trénovania nasledovné skóre:

Train accuracy: 0.9972 Train loss: 0.0082
Valid accuracy: 0.8073 Valid loss: 1.1078

Aj keď to nie je najhoršie skóre (minimálne sme porovnateľne lepší ako náhodný model), vzhľadom na vývoj skóre počas epoch sme presvedčení, že naša sieť sa pretrénováva. Počas jednotlivých epoch trénovacie skóre stúpa, kým validačné klesá.

Druhý baseline model, so snahou aspoň mierne zabrániť pretrénovaniu, dosiahol napokon podobné skóre:

Train accuracy: 1.0000 Train loss: 0.0081
Valid accuracy: 0.8062 Valid loss: 0.8747

Do budúcna chceme určite zvýšiť počet epoch trénovania a bojovať s pretrénovaním siete.

7. Trénovanie neurónky

Tréning našich neuróniek (rutina) začínala vždy definíciou modelu, následným nastavením hyper parametrov, a spustením tréningu (a logovaním výsledkov). Po natrénovaní modelu sme výsledky analyzovali z logov neurónky a nástrojom Tensorboard, kde sa skóre po jednotlivých epochách vykresľuje.

V ďalších iteráciach projektu sme vo štvrtom trénovanom modeli (model v4) nahradil ibidirectional vrstvu obyčajnou lstm vrstvou. Jeho dosiahnuté výsledky po 15 epochach boli:

Train accuracy: 0.9033 Train loss: 0.2419
Valid accuracy: 0.8368 Valid loss: 0.3846

Keďže sme videli, že valid loss stúpa, skúsili sme model nechať bežať pre 30 epoch (model v5). Jeho dosiahnuté výsledky boli:

Train accuracy: 0.9817 Train loss: 0.0649
Valid accuracy: 0.8272 Valid loss: 0.6638

Následne sme skúsili natrénovať model s bidirectional vrstvou, pre explicitne zadefinovanú forward aj backward vrstvu (model v6). Tento model dosiahol pri 30 epochách nasledujúce výsledky:

Train accuracy: 0.9782 Train loss: 0.0615
Valid accuracy: 0.8304 Valid loss: 0.7337

Pričom najlepšie hodnoty dosiahol tento model už po 14. Epoche

Train accuracy: 0.8787 Train loss: 0.2864
Valid accuracy: 0.8527 Valid loss: 0.3391

Následne sme skúsili trénovať model pre vyššiu lstm_size (model v7). Pre lstm_size zvýšenú na 128 dosiahol model po 30. epoche výsledky:

Train accuracy: 0.9919 Train loss: 0.0267
Valid accuracy: 0.8207 Valid loss: 1.0078

Zvýšenie lstm_size ešte zhoršilo pretrrénovanie modelu, preto sme skúsili hodnotu znížiť na 32 (model v8). Model po 30 epochách dosiahol nasledovné výsledky:

Train accuracy: 0.9393 Train loss: 0.1557
Valid accuracy: 0.8370 Valid loss: 0.4546

V tomto prípade nebolo nebolo pretrénovanie také výrazné, takže sme skúsili hodnotu znížiť na 16. Taktiež sme skúsili upraviť hodnotu learning_rate na 0.02 (model v9). Tento model bežal až pre 100 epoch, aby sme zistili ako sa bude model správať pri vyššom počte epoch.

Train accuracy: 0.9926 Train loss: 0.0255
Valid accuracy: 0.813 Valid loss: 1.1955

Po 100. Epochách bol model zbytočne pretrénovaný, keďže najlepšie výsledky dosiahol už po 15. Epoche:

Train accuracy: 0.8724 Train loss: 0.305
Valid accuracy: 0.8520 Valid loss: 0.3426

Ďalší model sme skúšali spustiť s upravenou hodnotou batch_size z 32 na 16, a znížením learning_rate na 0.01 (model v10). Po 30 epochách boli výsledky nasledovné:

Train accuracy: 0.9424 Train loss: 0.1491
Valid accuracy: 0.8269 Valid loss: 0.4756

V ďalšom modeli sme do bidirectional vrstvy pridali activity_regularizer l1 na boj s pretrénovaním (model v11). Avšak ani toto nám moc nepomohlo, keďže hodnoty po 30 epochách boli nasledovné:

Train accuracy: 0.9588 Train loss: 0.1230
Valid accuracy: 0.8222 Valid loss: 0.600

Najlepšie hodnoty dosiahol tento model po 12. Epoche

Train accuracy: 0.8788 Train loss: 0.2958
Valid accuracy: 0.8531 Valid loss: 0.3549

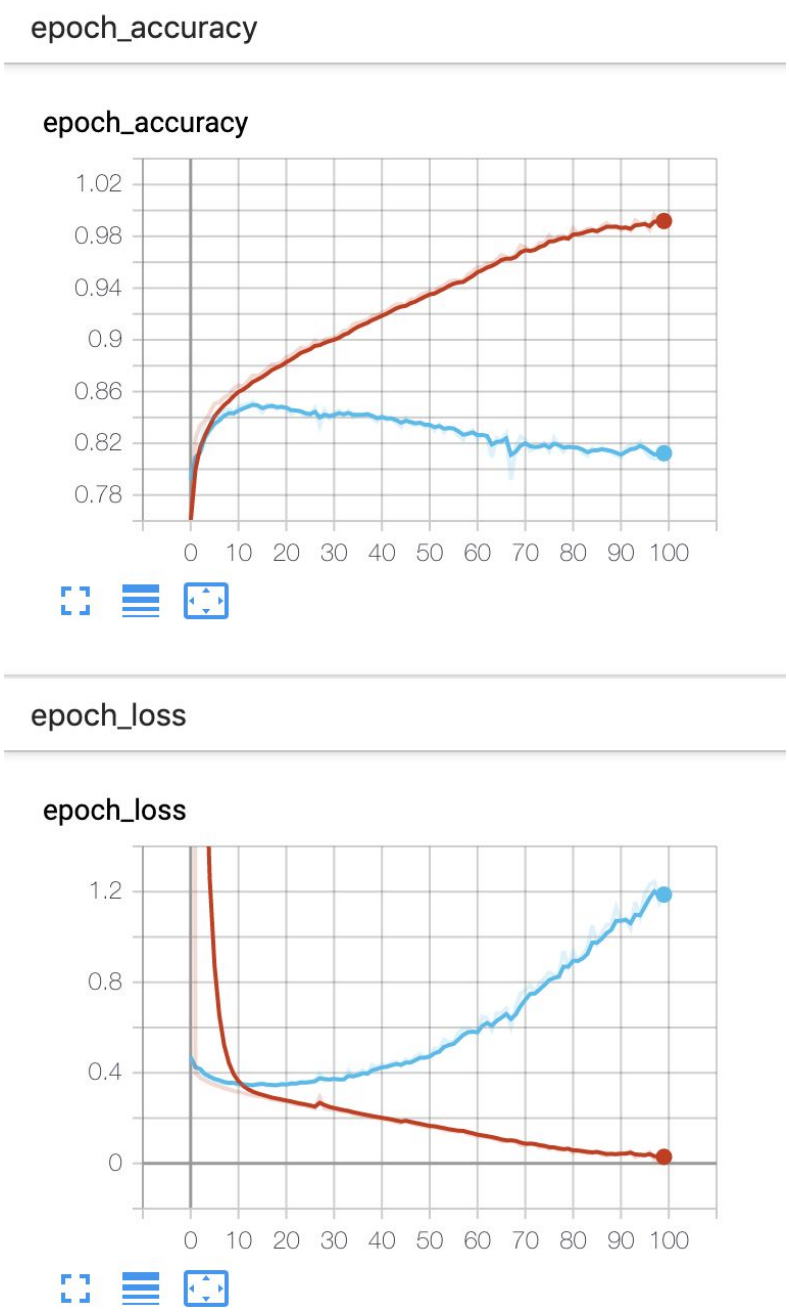
9. Vyhodnotenie výsledkov

Pre analýzu výsledkov sme využívali Tensorboard a logy samotného tréningu, na základe ktorých sme mohli jednotlivé neurónky porovnávať a vyhodnocovať. Pri vyhodnocovaní sme okrem samotných hodnôt, ktoré neurónka dosiahla, pozerali aj na ich vývoj (rast, klesanie) a stabilitu (napr. či posledných pár epoch sú hodnoty približne ustálené).

Najväčší problém, ktorý sprevádzal väčšinu našich tréningov, bolo overfitovanie neurónky - skóre na trénovacej vzorke sa nám darilo zlepšovať “až príliš”, pričom skóre na validačnej vzorke dát sa zhoršovalo. Proti overfittovaniu sme vyskúšali viaceré metódy - l1/l2 regularizáciu, pridanie dropout-u či použitie nami predtrénovaného embeddingu slov. Bohužiaľ, overfittovania sa nám do odovzdania projektu nepodarilo úplne zbaviť.

Po zavedení fasttext embeddingu a ponechani stopslov sa úspešnosť modelov podarilo zvýšiť približne na úroveň validačnej presnosti (accuracy) 0.84. Následné upravovanie hyperparametrov modelov už takéto zvýšenie neprinieslo. Pre rôzne hodnoty paramterov ako sú lstm_size, learning_rate, batch_size a podobne sa už výsledky príliš nelíšili. Modely vo všeobecnosti dosahovali najlepšie výsledky medzi 10. a 20. epochov a potom sa zlepšovali už len na trénovacej množine.

Celkovo najvyššiu validačnú presnosť dosiahol model s bidirectional vrstvou a dense vrstvou, obsahujúci activity_regularizer l1 s hodnotou 0.01, lstm_size = 32, batch_size = 32, learning_rate = 0.03 po 12. epoche behu a to s hodnotou **Valid accuracy: 0.8531.**



Obr2. Ukážka vyhodnotenia neurónky v prostrední Tensorboard (červená - trénovacia krivka, modrá-validačná. Na grafe vidieť efekt pretrénovania sa.)