Cel

Klasyfikacja (V+на+N4 r. żeń., l. p.) accusativ na frazem i nie frazem

Zasoby

Ręcznie annotowane 1087 trigramów w formie (V+на+N4 r. żeń., l. p.) accusativ z czego 1078 trigramów było unikalnych

Workflow

1. Preprocessing:

a) grupa-kontrolna.tsv (ręcznie annotowany zbiór)

- trigram był między '&&&' ,a '###'
- poprawione błędy w grupa-kontrolna.tsv typu
  - 4-gram zamiast 3-gram między '&&&', '###'
  - wyczyszczone niechciane znaki typu '(', '<<'

b) korpus ru.common-crawl.xz

- pierwotnie był już podzielony na zdania
- ściągnięty korpus common-crawl języka rosyjskiego, 105GB archiwum xz z statmt.org
- wszystkie liczby zamienione zostały na 'NUMBER', lowercasing, pozostawione tylko alphanumeryczne znaki

c) на-common-crawl.gz

- z wyczyszczonego archiwum korpusu commmon-crawl odfiltrowane zostały wszystkie zdania zawierające 'на' i spakowane do 74GB archiwum gzip

d) gazeta.txt

- skonkatenowane 3362 pliki rosyjskich gazet w txt w jeden 42MB plik txt

- Wytrenowany został PunktSentenceTokenizer z nltk.tokenize.punkt.Tokenizer na tym 42MB pliku.
- dokonana została segmentacja tego 42MB pliku
- wszystkie liczby zamienione zostały na 'NUMBER', lowercasing, pozostawione tylko alphanumeryczne znaki

PunktSentenceTokenizer

PunktSentenceTokenizer używa nienadzorowanego algrytmu Kiss & Strunk (2006)* do zbudowania modelu dla skrótów wyrazów, kolokacji i słów zaczynających zdanie. Potem używa tego modelu, aby znaleźć koniec i początek zdania. Wg. http://www.nltk.org/_modules/nltk/tokenize/punkt.html#PunktSentenceTokenizer to rozwiązanie działa dobrze dla wielu Eurpejskich języków

*więcej o Kiss & Strunk pod http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.85.5017&rep=rep1&type=pdf

e) Word2Vec

- wytrenowany na 38GB tekstu rosyjskiego + 42MB gazeta.txt
- trenowany na 10 cpu
- rozmiar wektorów 300
- zignorowane słowa, których liczba wynosi mniej niż 10
- użyty skip-gram model (więcej pod http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/)
- użyte negative sampling (więcej pod http://mccormickml.com/2017/01/11/word2vec-tutorial-part-2-negative-sampling/)
- maksymalny dystans między bieżącym, a przewidywanym słowem (w zdaniu) wynosi 5
- 5 iteracji na całym korpusie (5 epok)

f) grupa-kontrolna.cc.tsv (zbieranie nowych kontekstów dla trigramów)

- Skryptem w pythonie zostały wyekstrahowane z ha-commoncrawl.gz nowe konteksty dla trigramów z grupa-kontrolna.tsv
- wyszło 883877 nowych kontekstów dla trigramów
- po dodaniu ich do grupa-kontrolna.cc.tsv otrzymaliśmy ich 884965

h) Podział na zbiór uczący, testowy i developerski

- 63037 przykładów (w których było 49 unikalnych trigramów) zostało odłożone jako zbiór developerski, dev.tsv
- reszta została użyta do walidacji krzyżowej, i zapiszemy ją jako grupa-kontrolna.cc.minus-dev.tsv

2. Ekstrakcja cech

- lewy kontekst, 5 słów, każde z nich to wektor z Word2Vec'a
- prawy kontekst, 5 słów, każde z nich to wektor z Word2Vec'a
- trigram, każde z trzech słów to wektor z Word2Vec'a
- ostateczny feature vector to konkatenacja wektora lewego kontekstu, wektora prawego kontekstu i wektora trigramu

3. Uczenie

Wykorzystany został Stochastic Gradient Descent z regresją logistyczną. Po to aby móc trenować online. Było to niezbędne, bo wszystkie dane uczące nie mieściły się w pamięci na raz. Wykonana została jedna iteracja na całym korpusie.

Parametry dla SGDClassifier z regresją logistyczną:

- learning_rate='optimal' eta = 1.0 / (alpha * (t + t0)), gdzie eta to learning rate, a t0 jest wybierane na podstawie heurystyki od Leon Botto*
- penalty='elasticnet' (kombinacja wypukła L1 i L2)
- l1_ratio=0.5, (kontroluje kombinację wypukłą między regularyzacją L1, a regularyzacją L2)
- alpha=0.00015, (używane do obliczania learning_rate)

  *Więcej pod http://scikit-learn.org/stable/modules/sgd.html#sgd

4. Optymalizacja

- Dla każdej z cech (lewy konteks, trigram, prawy kontekst) sprawdzone zostały na pierwotnej grupa-kontrolna.tsv (1087 wierszy) następujące sposoby tworzenia wektorów, które po konkatenacji dawały ostateczny wektor cech:
  - konkatancja wektorów,
  - konkatenacja wektora z wektorem będącym minimum z wektorów słów i z wektorem będącym maximum z wektorów słów,
  - konkatenacja powyższego ze średnią wektorów słów,

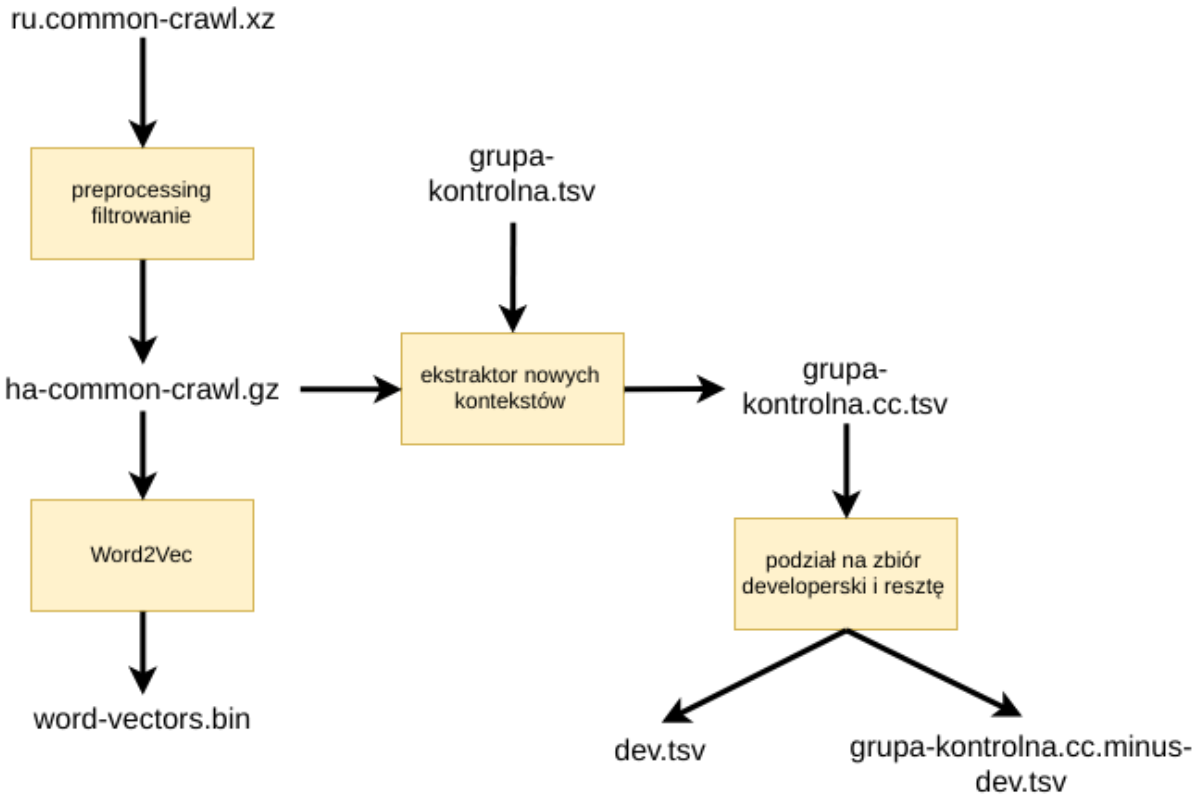
    - suma wektorów
    - średnia wektorów
    - minimum z wektorów słów
    - maximum z wektorów słów
- okazało się, że najlepiej dla każdej z cech sprawdza się konkatenacja wektorów słów
- na zbiorze developerskim została dobrana najlepsza alfa, penalty i l1_ratio

5. Ewaluacja

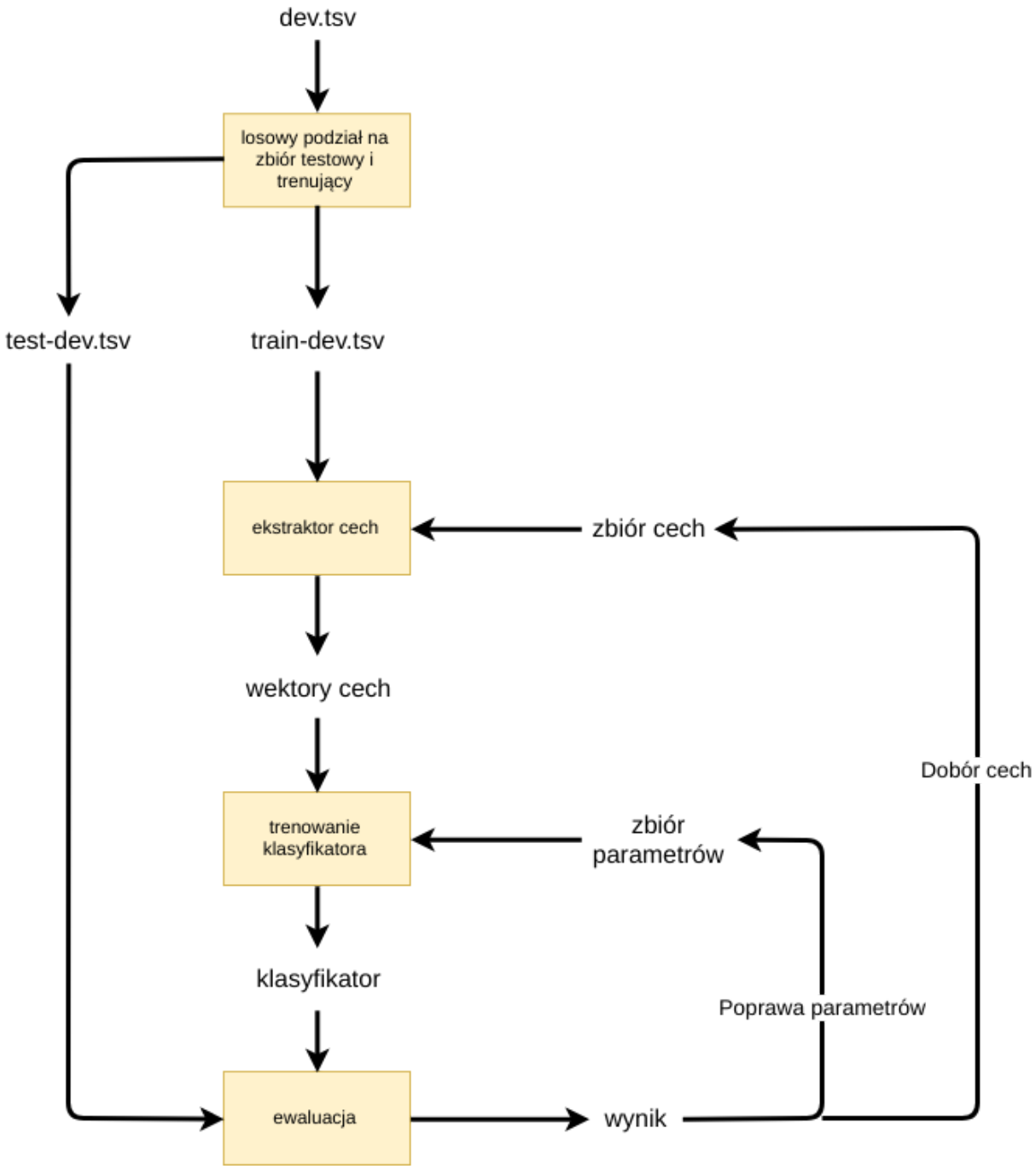
- metodą Monte-Carlo cross-validation (reapeated random sub-sampling validation)
- Za każdym razem grupa-kontrolna.cc.minus-dev.tsv była losowo dzielona na zbiór testowy i zbiór uczący przy czym zadbane było, aby żadne dwa identyczne trigramy o różnych kontekstach nie znalazły się jednocześnie w zbiorze testowym i zbiorze uczącym. Zbiór testowy średnio był 10 krotnie mniejszy niż zbiór uczący.
- Zbiór uczący był następnie powiększany o zbiór developerski i na tym powiększonym zbiorze trenowany był klasyfikator, a walidowany był na zbiorze testowym.
- Proces został powtórzony 250 razy, a ostateczna dokładność była średnią dokładnością z tych 250.
- Wynik: 0.72 (72% dokładności), poprzedni wynik na samym grupa-kontrolna.tsv bez dodatowych kontekstów z common crawl'a wyniósł 0.665

Zrzuty ekranu

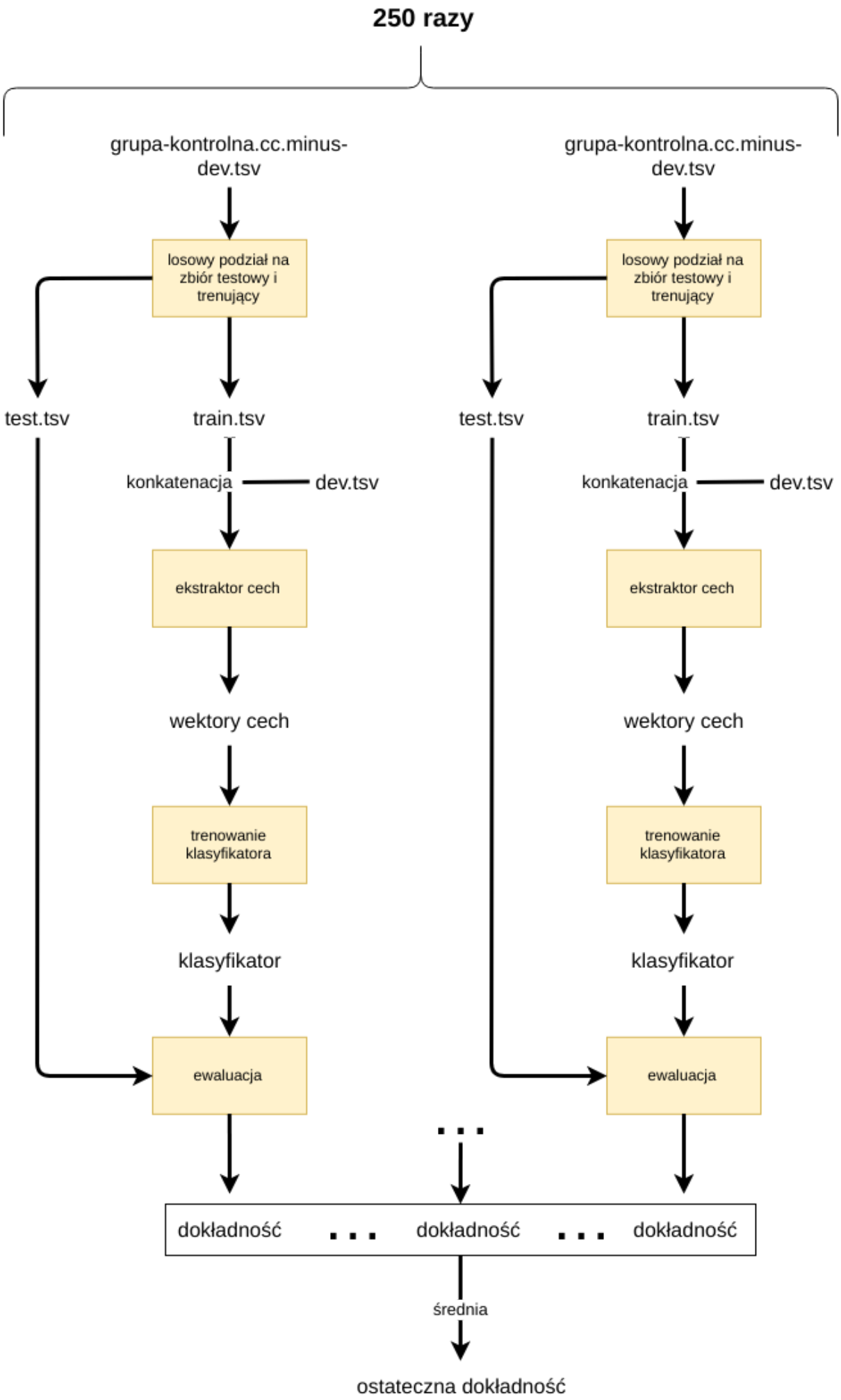
Przygotowanie zbiorów i wektorów słów



Optymalizacja


dev.tsv
losowy podział na zbiór testowy i trenujący
test-dev.tsv
train-dev.tsv
ekstraktor cech
zbiór cech
wektory cech
Dobór cech
trenowanie klasyfikatora
zbiór parametrów
klasyfikator
Poprawa parametrów
ewaluacja
wynik

Ewaluacja

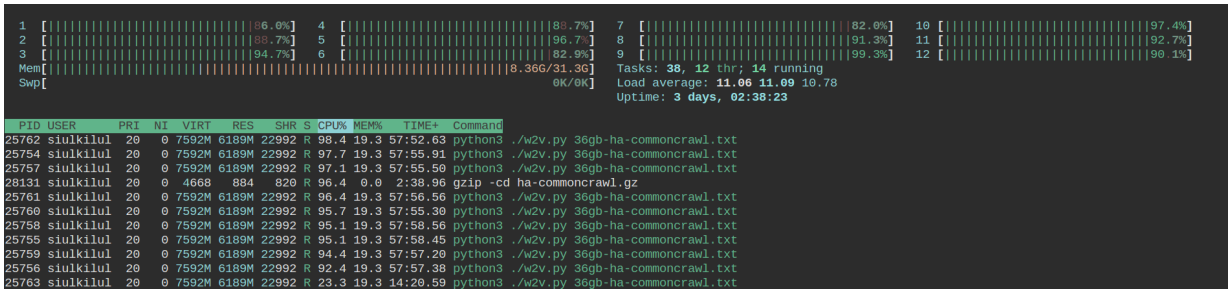

250 razy
grupa-kontrolna.cc.minus-dev.tsv
losowy podział na zbiór testowy i trenujący
test.tsv
train.tsv
konkatenacja
dev.tsv
ekstraktor cech
wektory cech
trenowanie klasyfikatora
klasyfikator
ewaluacja
grupa-kontrolna.cc.minus-dev.tsv
losowy podział na zbiór testowy i trenujący
test.tsv
train.tsv
konkatenacja
dev.tsv
ekstraktor cech
wektory cech
trenowanie klasyfikatora
klasyfikator
ewaluacja
dokładność
dokładność
dokładność
średnia
ostateczna dokładność

Trenowanie Word2Vec

```
[siulkilulki@siulkilulki yf]$ head -n 20 log-80
2017-10-29 17:41:20,276 : INFO : Mulitcore version. Good to go.
2017-10-29 17:41:20,276 : INFO : collecting all words and their counts
2017-10-29 17:41:20,282 : INFO : PROGRESS: at sentence #0, processed 0 words, keeping 0 word types
2017-10-29 17:41:20,429 : INFO : PROGRESS: at sentence #10000, processed 218567 words, keeping 49131 word types
2017-10-29 17:41:20,576 : INFO : PROGRESS: at sentence #20000, processed 442346 words, keeping 76287 word types
2017-10-29 17:41:20,729 : INFO : PROGRESS: at sentence #30000, processed 672074 words, keeping 97705 word types
2017-10-29 17:41:20,887 : INFO : PROGRESS: at sentence #40000, processed 894623 words, keeping 114714 word types
2017-10-29 17:41:21,104 : INFO : PROGRESS: at sentence #50000, processed 1272898 words, keeping 144654 word types
2017-10-29 17:41:21,303 : INFO : PROGRESS: at sentence #60000, processed 1630226 words, keeping 168785 word types
2017-10-29 17:41:21,522 : INFO : PROGRESS: at sentence #70000, processed 2021305 words, keeping 192433 word types
2017-10-29 17:41:21,742 : INFO : PROGRESS: at sentence #80000, processed 2425084 words, keeping 215257 word types
2017-10-29 17:41:21,977 : INFO : PROGRESS: at sentence #90000, processed 2846055 words, keeping 237656 word types
2017-10-29 17:41:22,224 : INFO : PROGRESS: at sentence #100000, processed 3274710 words, keeping 257962 word types
2017-10-29 17:41:22,473 : INFO : PROGRESS: at sentence #110000, processed 3705930 words, keeping 276010 word types
```

```
2017-10-28 22:21:32,585 : INFO : PROGRESS: at sentence #59990000, processed 3237864124 words, keeping 15188108 word types
2017-10-28 22:21:32,942 : INFO : PROGRESS: at sentence #60000000, processed 3238418550 words, keeping 15189923 word types
2017-10-28 22:21:33,087 : INFO : collected 15189934 word types from a corpus of 3238428829 raw words and 60000192 sentences
2017-10-28 22:21:33,088 : INFO : Loading a fresh vocabulary
2017-10-28 22:21:42,621 : INFO : min_count=10 retains 1785468 unique words (11% of original 15189934, drops 13404466)
2017-10-28 22:21:42,621 : INFO : min_count=10 leaves 3214783956 word corpus (99% of original 3238428829, drops 23644873)
2017-10-28 22:21:47,304 : INFO : deleting the raw counts dictionary of 15189934 items
2017-10-28 22:21:47,777 : INFO : sample=0.001 downsamples 26 most-common words
2017-10-28 22:21:47,777 : INFO : downsampling leaves estimated 2750436646 word corpus (85.6% of prior 3214783956)
2017-10-28 22:21:47,777 : INFO : estimated required memory for 1785468 words and 300 dimensions: 5177857200 bytes
2017-10-28 22:21:54,261 : INFO : resetting layer weights
2017-10-28 22:22:14,619 : INFO : training model with 10 workers on 1785468 vocabulary and 300 features, using sg=1 hs=0 sample=0.001 negative=5 window=5
2017-10-28 22:22:15,634 : INFO : PROGRESS: at 0.00% examples, 191901 words/s, in_qsize 17, out_qsize 1
2017-10-28 22:22:16,645 : INFO : PROGRESS: at 0.01% examples, 263105 words/s, in_qsize 18, out_qsize 0
2017-10-28 22:22:17,655 : INFO : PROGRESS: at 0.02% examples, 296568 words/s, in_qsize 16, out_qsize 0
2017-10-28 22:22:18,689 : INFO : PROGRESS: at 0.02% examples, 301133 words/s, in_qsize 20, out_qsize 0
```

```
2017-10-29 08:38:40,444 : INFO : PROGRESS: at 100.00% examples, 338825 words/s, in_qsize 19, out_qsize 1
2017-10-29 08:38:41,447 : INFO : PROGRESS: at 100.00% examples, 338826 words/s, in_qsize 16, out_qsize 0
2017-10-29 08:38:42,041 : INFO : worker thread finished; awaiting finish of 9 more threads
2017-10-29 08:38:42,089 : INFO : worker thread finished; awaiting finish of 8 more threads
2017-10-29 08:38:42,097 : INFO : worker thread finished; awaiting finish of 7 more threads
2017-10-29 08:38:42,104 : INFO : worker thread finished; awaiting finish of 6 more threads
2017-10-29 08:38:42,116 : INFO : worker thread finished; awaiting finish of 5 more threads
2017-10-29 08:38:42,152 : INFO : worker thread finished; awaiting finish of 4 more threads
2017-10-29 08:38:42,165 : INFO : worker thread finished; awaiting finish of 3 more threads
2017-10-29 08:38:42,177 : INFO : worker thread finished; awaiting finish of 2 more threads
2017-10-29 08:38:42,180 : INFO : worker thread finished; awaiting finish of 1 more threads
2017-10-29 08:38:42,185 : INFO : worker thread finished; awaiting finish of 0 more threads
2017-10-29 08:38:42,185 : INFO : training on 16192144145 raw words (13752151106 effective words) took 40587.6s, 338827 effective words/s
2017-10-29 08:38:42,187 : INFO : saving Word2Vec object under ru_crawl.38gb.bin, separately None
2017-10-29 08:38:42,187 : INFO : storing np array 'syn0' to ru_crawl.38gb.bin.wv.syn0.npy
2017-10-29 08:38:43,100 : INFO : not storing attribute syn0norm
2017-10-29 08:38:43,100 : INFO : storing np array 'syn1neg' to ru_crawl.38gb.bin.syn1neg.npy
2017-10-29 08:38:44,030 : INFO : not storing attribute cum_table
2017-10-29 08:38:48,075 : INFO : saved ru_crawl.38gb.bin
```



Jeden z pierwszych protypów

```
LogisticRegressionCV(Cs=10, class_weight=None, cv=None, dual=False,
           fit_intercept=True, intercept_scaling=1.0, max_iter=100,
           multi_class='ovr', n_jobs=1, penalty='l2', random_state=None,
           refit=True, scoring=None, solver='liblinear', tol=0.0001,
           verbose=0)
Score: 0.6651833433364773, func-1: concat, func-2: concat, window: 1
Score: 0.6651833433364773, func-1: concat, func-2: sum_word_vectors, window: 1
Score: 0.6651833433364773, func-1: concat, func-2: mean, window: 1
Score: 0.6651833433364773, func-1: concat, func-2: _min, window: 1
Score: 0.6651833433364773, func-1: concat, func-2: _max, window: 1
Score: 0.6606054760067449, func-1: concat, func-2: _max, window: 4
Score: 0.6504448223409897, func-1: concat, func-2: sum_min_max, window: 3
Score: 0.6487211853884236, func-1: concat, func-2: min_max_avg_concat, window: 4
Score: 0.6487126906551584, func-1: concat, func-2: min_max_concat, window: 4
Score: 0.648687206455362, func-1: concat, func-2: min_max_concat, window: 5
```

Ewaluacja

```
(yf) [siulkilulki@siulkilulki yf]$ ./split_data.py grupa-kontrolna-merged.counted.minus-dev.fixed.tsv train.tsv test.tsv
2017-11-15 23:52:19,103 : INFO : Splitting data.
2017-11-15 23:52:19,103 : INFO : Building dicts.
2017-11-15 23:52:26,760 : INFO : Oversampling.
2017-11-15 23:52:26,762 : INFO : it 15, balance: 0
2017-11-15 23:52:28,025 : INFO : it 2, balance: 0
Train set rows: 1404457 Test set rows: 92835
2017-11-15 23:52:28,103 : INFO : Train set rows: 1404457        Test set rows: 92835
2017-11-15 23:52:28,103 : INFO : Saving train.tsv.
2017-11-15 23:52:32,952 : INFO : Saving test.tsv.
(yf) [siulkilulki@siulkilulki yf]$ cat train.tsv <(tail -n +2 DEV.tsv) > train-plus-dev.tsv
(yf) [siulkilulki@siulkilulki yf]$ ./train.py train-plus-dev.tsv DEV.tsv test.tsv ru_crawl.38gb.bin
INFO:Generating features
INFO:loading Word2Vec object from ru_crawl.38gb.bin
INFO:loading wv recursively from ru_crawl.38gb.bin.wv.* with mmap=None
INFO:loading syn0 from ru_crawl.38gb.bin.wv.syn0.npy with mmap=None
INFO:setting ignored attribute syn0norm to None
INFO:loading syn1neg from ru_crawl.38gb.bin.syn1neg.npy with mmap=None
INFO:setting ignored attribute cum_table to None
INFO:loaded ru_crawl.38gb.bin
INFO:reading table
INFO:DONE
INFO:preparing Q3
INFO:DONE
INFO:preparing zdanie
INFO:DONE
INFO:Shuffling
INFO:Chunk: 1, Partial fit
INFO:Partial fit for chunk 1 DONE
INFO:preparing Q3
INFO:DONE
INFO:preparing zdanie
INFO:DONE
INFO:Shuffling
INFO:Chunk: 2, Partial fit
```