نکات/نیازمندها برای اجرای کدها

برای اینکه بتوان کدهای آپلود شده را اجرا کرد باید پوشه‌ی CA1_Data در کنار پوشه‌ی Code قرار بگیرد.
برای اجرای کدها کافی است فایل main.py را اجرا کرده و کد مربوط به تمامی سوالات یکی پس از دیگری اجرا خواهد شد.

دادگان

همان طور که در صورت سوال بیان شده است بخشی از دادگان از کل داده‌ها جدا شده و در اختیار ما قرار گرفته است. در این بخش به بررسی داده‌ها می‌پردازیم.
برای کار با این داده‌ها فایلی با نام 0_data_readers.py نوشته شده است که ۲ تابع زیر در آن تعریف شده است:

```
1. read_the_data(data_type)
2. read_file_content(file_path)
3. first_look_at_the_data()
```

تابع (۱) وظیفه‌ی خواندن داده‌ها را دارد که در سوال‌های بعدی مورد استفاده قرار می‌گیرد.
آمار مربوط به داده‌ها به شرح زیر است:

Feature	Value
Number of Train instances	2381
Number of Test instances	600
Average Character Length of Train Instances	2052.25
Average Character Length of Test Instances	4548.55

به طور کلی علاقه‌مندیم که داده‌های تست و آموزش از توزیع یکسانی باشند تا بتوانیم موارد یادگرفته شده روی داده‌های آموزش را به داده‌های تست تعمیم دهیم. با همین آمار ساده می‌توان دید که متوسط طول اخبار داده‌های تست بیشتر از دو برابر داده‌های آموزش است با این وجود اگر متوسط طول کمتر بود بیشتر نگران می‌شدیم چون معمولا یادگیری روی متون کوتاه دشوارتر از یادگیری روی متون طولانی‌تر است چون هر چه متن طولانی‌تر باشد اطلاعات بیشتری برای استخراج دارد. در نتیجه خیلی نگران بیشتر بودن طول داده‌های تست نمی‌شویم چون احتمالا اطلاعات بیشتری می‌توان از آن استخراج کرد ولی باید توجه داشته باشیم که حداقل از نظر طول اخبار توزیع‌ها کاملا یکی نیستند.

خروجی کد به شکل زیر است:

```
##################### Data #####################
Number of training data: 2381
Number of test data: 600
Avg. Character Length of Train instances: 2052.2507349853004
Avg. Character Length of Test instances: 4548.551666666666
```

سوال ۱ - پیش‌پردازش

کد این سوال در فایل q1.py موجود است.
برای مشخص کردن گام‌های مورد نیاز در پیش‌پردازش متن از مراجع (۱-۵) کمک می‌گیریم و با توجه به این مراجع تلاش می‌کنیم گام‌های زیر را برای پیش‌پردازش متون اجرا کنیم:
۱) درست کردن فاصله‌ها و تاریخ‌ها در جمله

اولین گام پیشنهاد شده در (۲) درست کردن فاصله‌های موجود در متن است و به این شکل بیان شده است که در این گام مطمئن می‌شویم نیم‌فاصله‌ها رعایت شده و فاصله‌های اضافی حذف می‌شوند براین این کار از تابع normalize از کتابخانه‌ی هضم (۶) را استفاده می‌کنیم که هدف این تابع به طور خلاصه درست کردن نیم‌فاصله‌ها است. اما این کتابخانه تمام موارد را تشخیص نمی‌دهد به همین دلیل از کتابخانه‌ی ویراستار (۷) و Parsivar (۸) نیز در این گام کمک می‌گیریم.

۲) قطعه‌بندی جملات

پس از نرمالایز کردن متن به سراغ جداسازی جملات می‌رویم تا متن را به تعداد جمله‌ی جدا از هم تبدیل کنیم. برای این کار از کتابخانه‌ی Parsivar کمک می‌گیریم.

۳) قطعه‌بندی کلمات

در گام بعدی بر روی هر یک از جملات تابع جداسازی کلمات جمله را اجرا می‌کنیم تا به این شکل token بندی کلمات انجام شود.

۴) lemmatization برای هر یک از کلمات

علت استفاده از lemmatization این است که می‌خواهیم ریشه‌ی کلمات را بدست آوریم. به عنوان مثال در خروجی این بخش فعل‌ها به ریشه‌ی اصلی خود تغییر پیدا کرده‌اند.

گام stemming را انجام نمی‌دهیم چون با اینکه این کار می‌تواند مواردی مانند کتاب‌ها را به کتاب تبدیل کند اما شکل ظاهری تعداد بسیار زیادی از کلمات را به هم می‌ریزد و به عنوان مثال «مقامات» را به «مقا» تبدیل می‌کند که مطلوب نیست.

سوال ۲ - مدل زبانی

در این سوال مدل‌های زبانی خواسته شده را پیاده‌سازی می‌کنیم. کد این سوال در فایل q2.py موجود است.
به طور کلی برای پیاده سازی همان طور که در صورت سوال راهنمایی شده است از nltk و به طور خاص بخش language model آن استفاده می‌کنیم. این کتابخانه توابع لازم برای محاسبه‌ی ngram ها و fit کردن مدل را در اختیار ما قرار می‌دهد. نکاتی که در حین انجام این کار خوب است که به آن‌ها توجه کنیم عبارتند از:

- جملات را pad کنیم یعنی کاراکترهای شروع و پایان جمله را به آن‌ها اضافه کنیم، این کاراکترها در محاسبه‌ی احتمالات به خصوص در مدل bigram کلیدی هستند.
- برای مدل‌های character level زمانی که داریم کاراکترها را در کنار هم قرار می‌دهیم تا ngramهای مورد نظر را استخراج کنیم دقت کنیم که بین کلمات مختلف علامت فاصله قرار داده شود. این مسئله در مدل‌های word level اهمیتی ندارد چون به صورت پیش‌فرض می‌دانیم که مابین هر دو کلمه فاصله قرار دارد ولی این مسئله برای کاراکترها صادق نیست و مدل زبانی باید یادبگیرد که آیا بین این دو کاراکتر فاصله هست یا نه.

چند نتیجه از کد را در اینجا می‌آورم:
۱) برای جداسازی داده‌های آموزش و تست به این شکل عمل می‌کنم که هر خبر به احتمال ۸۰٪ در دسته‌ی آموزش قرار می‌گیرد و برای این کار نیز از تولید اعداد تصادفی استفاده می‌کنم. این کار باعث می‌شود که دسته‌های ایجاد شده واقعا تصادفی باشند. نتیجه‌ی یکی از دفعات اجرای کد به شکل زیر است:

```
Train Count: 1865
Train Examples in each Category:
     {
          'politic': 318,
          'technology': 227,
          'sport': 342,
          'social': 319,
          'finance': 322,
          'culture': 337
     }
Test Count: 516
```

۲) تعداد کلمات یکتا (vocabulary) تشخیص داده شده برای word level bigram برابر با ۱۵۱۸۳ است. (این عدد یکتا نیست و همان طور که بالاتر بیان شد این نتایج فقط برای نشان دادن کارکرد کد و برای یکی از دفعات اجرا می‌باشند.)

۳) یک نمونه از bigramها برای بخشی از یک خبر به شکل زیر است:

```
[('<s>', 'ان'), ('ان', 'که'), ('که', 'گونه'), ('گونه', 'ان'),
('ان', 'روزنامه'), ('روزنامه', 'تایمز'), ('تایمز', 'مسکو'),
('مسکو', 'روزنامه'), ('روزنامه', 'نوشته'), ('نوشته', 'برخی'),
('برخی', 'مشاوران'), ('مشاوران', 'وزارت'), ('وزارت', 'دفاع'),
('دفاع', 'این'), ('این', 'کشور'), ('کشور', 'مدعی'), ('مدعی', 'هستند'),
('هستند', 'که'), ('که', 'این'), ('این', 'کشور'), ('کشور', 'برای'),
('برای', 'مقابله'), ('مقابله', 'با'), ('با', 'امریکا'), ('امریکا', 'در'),
('در', 'اوکراین'), …
```

سوال ۳ - معیار سرگشتگی

الف) در این بخش به گزارش دقت و معیارهای میزان مطلوب بودن هر یک از ۴ مدل می‌پردازیم:

|> Testing the Word Unigram Model

```
Accuracy: 0.777319587628866
Recall (weighted): 0.777319587628866
F1 (macro): 0.7701430550867432
F1 (micro): 0.777319587628866
F1 (weighted): 0.7730831199949978
Report:               precision    recall  f1-score   support

     politic       0.73      0.67      0.70        84
  technology       0.71      0.84      0.77        89
       sport       0.75      0.69      0.72        83
      social       0.65      0.53      0.59        73
     finance       0.99      0.97      0.98        90
     culture       0.81      0.95      0.88        66

    accuracy                           0.78       485
   macro avg       0.77      0.78      0.77       485
weighted avg       0.78      0.78      0.77       485
```

|> Testing the Word Bigram Model

```
Accuracy: 0.1711340206185567
Recall (weighted): 0.1711340206185567
F1 (macro): 0.04870892018779343
F1 (micro): 0.1711340206185567
F1 (weighted): 0.05001452011035284
Report:               precision    recall  f1-score   support

     politic       0.00      0.00      0.00        84
  technology       0.00      0.00      0.00        89
       sport       0.17      1.00      0.29        83
      social       0.00      0.00      0.00        73
     finance       0.00      0.00      0.00        90
     culture       0.00      0.00      0.00        66

    accuracy                           0.17       485
   macro avg       0.03      0.17      0.05       485
weighted avg       0.03      0.17      0.05       485
```

|> Testing the Character Unigram Model

```
Accuracy: 0.20412371134020618
Recall (weighted): 0.20412371134020618
F1 (macro): 0.17051762588094768
F1 (micro): 0.20412371134020618
F1 (weighted): 0.16181626964347331
Report:               precision    recall  f1-score   support
```

```
     politic       0.00      0.00      0.00        84
  technology       0.00      0.00      0.00        89
       sport       0.18      0.14      0.16        83
      social       0.16      0.38      0.22        73
     finance       0.20      0.44      0.28        90
     culture       0.49      0.29      0.36        66

    accuracy                           0.20       485
   macro avg       0.17      0.21      0.17       485
weighted avg       0.16      0.20      0.16       485
```

|> Testing the Character Bigram Model

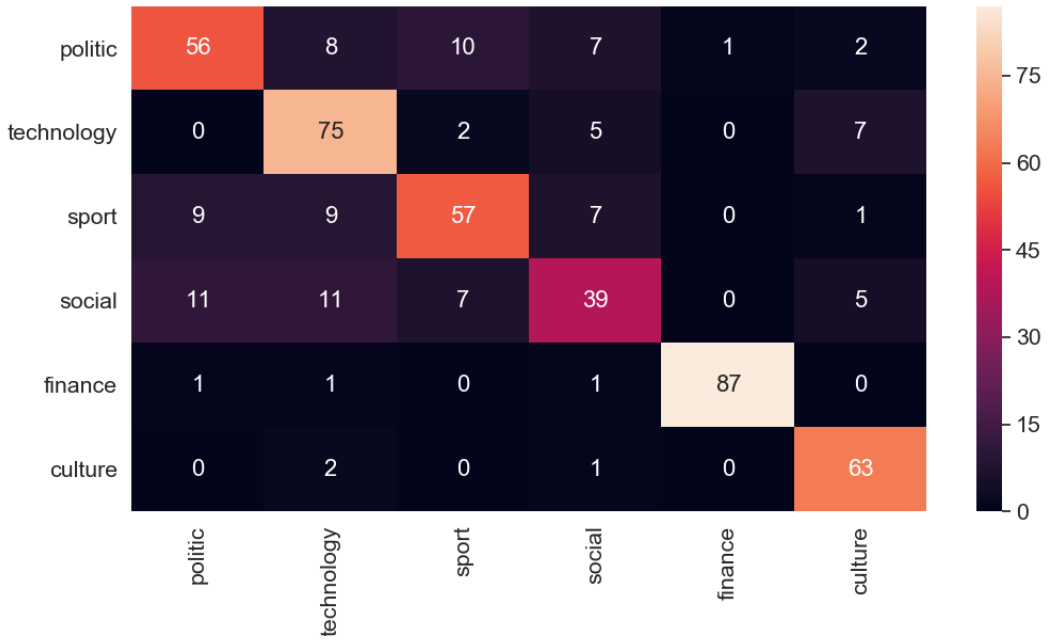
```
Accuracy: 0.1711340206185567
Recall (weighted): 0.1711340206185567
F1 (macro): 0.04870892018779343
F1 (micro): 0.1711340206185567
F1 (weighted): 0.05001452011035284
Report:               precision    recall  f1-score   support

     politic       0.00      0.00      0.00        84
  technology       0.00      0.00      0.00        89
       sport       0.17      1.00      0.29        83
      social       0.00      0.00      0.00        73
     finance       0.00      0.00      0.00        90
     culture       0.00      0.00      0.00        66

    accuracy                           0.17       485
   macro avg       0.03      0.17      0.05       485
weighted avg       0.03      0.17      0.05       485
```

با توجه به این نتایج می‌توان دید که بهترین مدل Word Level Unigram است. می‌توان دید که این مدل به طور چشم‌گیری از سایر مدل‌ها بهتر است. علت این است که:

- مدل‌های character level به این دلیل خوب عمل نمی‌کند این است که کاراکترها مخصوصا در زبان فارسی الزاما در کنار کاراکترهای دیگری نمی‌آیند و ترکیب‌های بسیار متفاوتی برای هر کاراکتر ممکن است به همین دلیل پیش‌بینی کاراکتر بعدی دشوار است.
- علت اینکه bigramها خوب عمل نمی‌کنند احتمالا این است که sparsity داده‌ها بالا است و ترکیب‌های دوتایی داده‌ها کم دیده می‌شوند و به همین علت احتمالات اعداد کوچکی می‌شوند. این که این ترکیب‌ها کم هستند نیز خیلی دور از ذهن نیست چون اسم‌های خاص به خصوص در اخبار زیاد است و در نتیجه ممکن است نام هر یک از این مکان‌ها یا اشخاص تنها یک یا دو بار دیده می‌شوند و به همین دلیل محاسبه‌ی احتمالات کم می‌شود. (۹ - ۱۰)

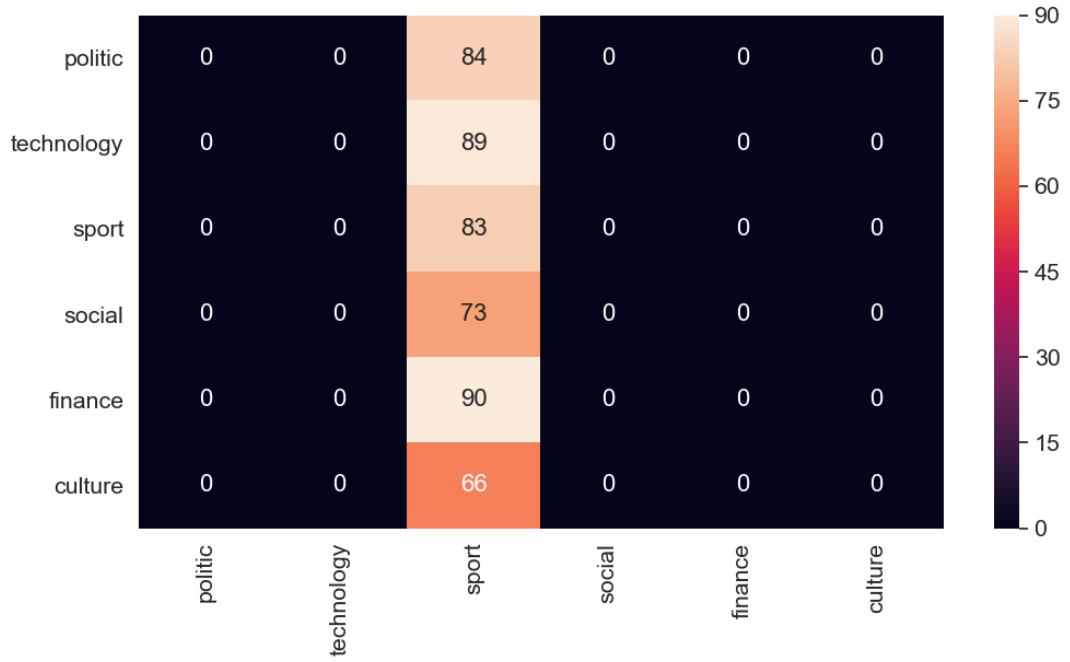
ب) معیارهای مدل‌ها در بخش «الف» گزارش شده است. ماتریس‌های درهم ریختگی برای هر یک از ۴ مدل به شکل زیر هستند. (زیر هر عکس نام آن به صورت caption مشخص شده است) می‌توان دید که مدل Unigram (Word Level) خوب تشخیص می‌دهند اما مدل‌های دیگر به خصوص مدل‌های bigram همه‌ی داده‌ها را در یک کلاس قرار می‌دهند. (در کلاس ورزش)

	politic	technology	sport	social	finance	culture
politic	56	8	10	7	1	2
technology	0	75	2	5	0	7
sport	9	9	57	7	0	1
social	11	11	7	39	0	5
finance	1	1	0	1	87	0
culture	0	2	0	1	0	63



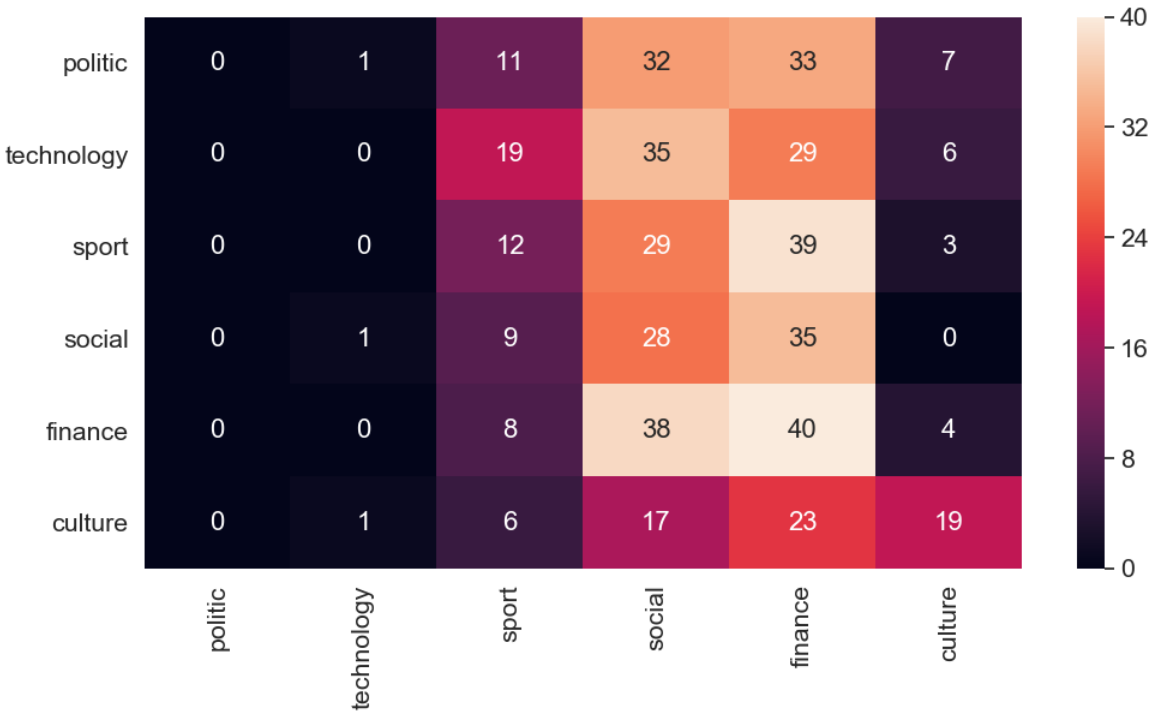
Unigram Word Confusion Matrix

	politic	technology	sport	social	finance	culture
politic	0	0	84	0	0	0
technology	0	0	89	0	0	0
sport	0	0	83	0	0	0
social	0	0	73	0	0	0
finance	0	0	90	0	0	0
culture	0	0	66	0	0	0



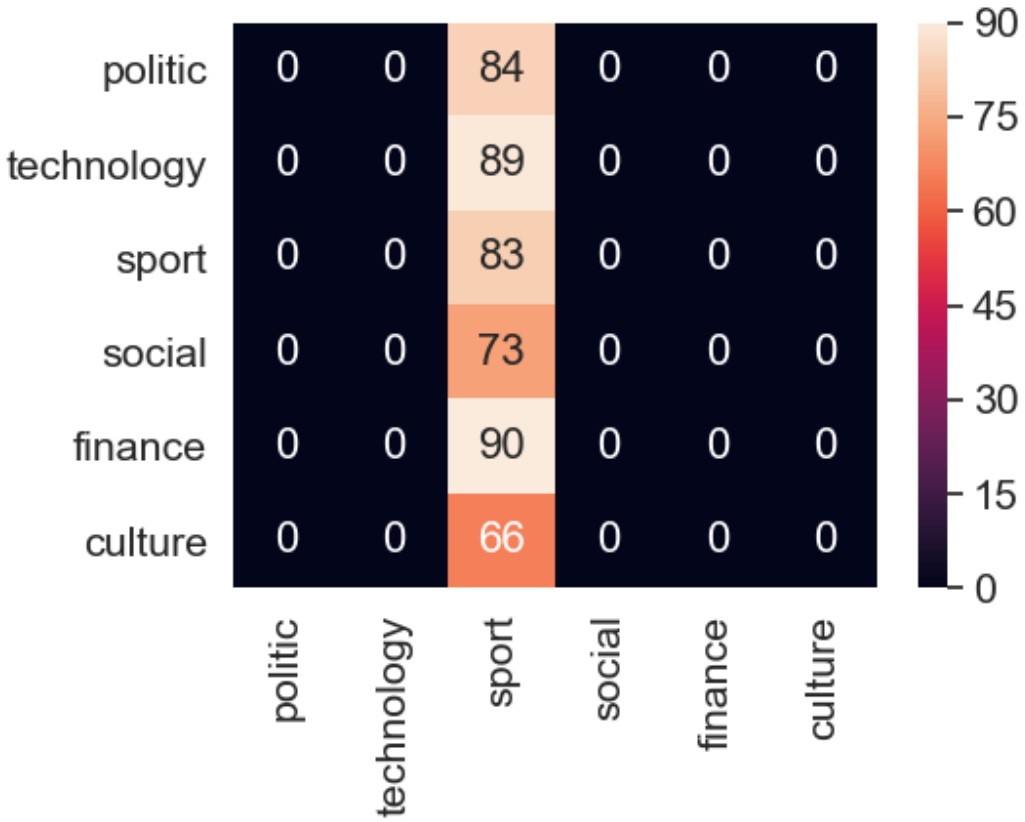
Bigram Word Confusion Matrix

	politic	technology	sport	social	finance	culture
politic	0	1	11	32	33	7
technology	0	0	19	35	29	6
sport	0	0	12	29	39	3
social	0	1	9	28	35	0
finance	0	0	8	38	40	4
culture	0	1	6	17	23	19



Unigram Character Confusion Matrix

	politic	technology	sport	social	finance	culture
politic	0	0	84	0	0	0
technology	0	0	89	0	0	0
sport	0	0	83	0	0	0
social	0	0	73	0	0	0
finance	0	0	90	0	0	0
culture	0	0	66	0	0	0



Bigram Character Confusion Matrix

سوال ۴ - دادگان تست

بهترین مدل زبانی مدل Unigram - Word level است که نتایج را به روش بیان شده در بخش «راهنمایی سوال ۳» با این مدل برای داده‌های تست تولید می‌کنیم و نتایج را در Results.csv می‌نویسیم. باید دقت شود که داده‌های تست و آموزش هر دو قبل از کار تمیز شوند.

مراجع

۱- اسلاید "Basic Text Processing" (اسلاید شماره دو درس)
2- Mohtaj, Salar, et al. "Parsivar: A language processing toolkit for persian." Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). 2018.
3- Shamsfard, Mehrnoush, Soheila Kiani, and Yaseer Shahedi. "STeP-1: standard text preparation for Persian language." *Third Workshop on Computational Approaches to Arabic Script-based Languages*. 2009.
4- https://towardsdatascience.com/nlp-text-preprocessing-a-practical-guide-and-template-d80874676e79, Last accessed on: February 27, 2020
5- https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html, Last accessed on: February 27, 2020

۶- کتابخانه‌ی هضم: http://www.sobhe.ir/hazm/
۷- کد/کتابخانه‌ی ویراستار: https://github.com/JKhakpour/virastar.py
۸- کتابخانه‌ی پارسی‌وار: https://github.com/ICTRC/Parsivar

9- https://stackoverflow.com/questions/36542993/when-are-uni-grams-more-suitable-than-bi-grams-or-higher-n-grams
10- https://stackoverflow.com/questions/43463792/what-is-the-difference-between-bigram-and-unigram-text-features-extraction