Author Verification Using Common N-Gram Profiles of Text Documents

Над проектом работали:
Алла Горбунова, Лика Джиоева, Евгения Егорова,
Елизавета Клыкова и Яна Шишкина

На основе статьи Magdalena Jankowska, Evangelos Milios & Vlado Kešelj (2014)

Датасет

- Оригинальный датасет соревнования PAN 2013 для Authorship Verification Task (train + test)
- Часть для обучения: 10 наборов для английского, 20 для греческого, 5 для испанского, в каждом наборе 1-10 документов известного автора и один -- неизвестного; + файл с ответами
- Тестовая часть: по 30 наборов для английского и греческого, 25 для испанского + файл с ответами
- Средняя длина документа ок. 1200 слов

Предобработка текстов

- Считывание текстов в pandas датафрейм с информацией о документах (автор, язык, текст и его длина)
- Токенизация с помощью NLTK, очистка от пунктуации
- Получение списков токенов и слов
- Составление n-грамм с помощью библиотеки NLTK
- Несколько видов n-грамм: по 1-3 слова и по 3-10 символов
- Добавление нормализованных частотностей n-грамм
- Словные n-граммы составлялись из списка слов, символьные -- из сырого текста

Создание профилей

- Профили имеют динамическую длину (для каждого типа n-грамм она выбирается отдельно -- профили обрезаются по самому короткому)
- При этом для всех текстов профили определенного типа имеют одинаковую длину
- Если при сравнении профилей выясняется, что в одном из них нет определенных n-грамм, эти n-граммы добавляются в профиль с частотностью 0

profile P_1	
n-gram	normalized frequency f_1
_ t h e	0.0127
t h e _	0.0098
a n d _	0.0052
_ a n d	0.0049
i n g _	0.0047
_ t o _	0.0044

profile P_2	
n-gram	normalized frequency f_2
_ t h e	0.0148
t h e _	0.0115
a n d _	0.0053
_ o f _	0.0052
_ a n d	0.0052
i n g _	0.0040

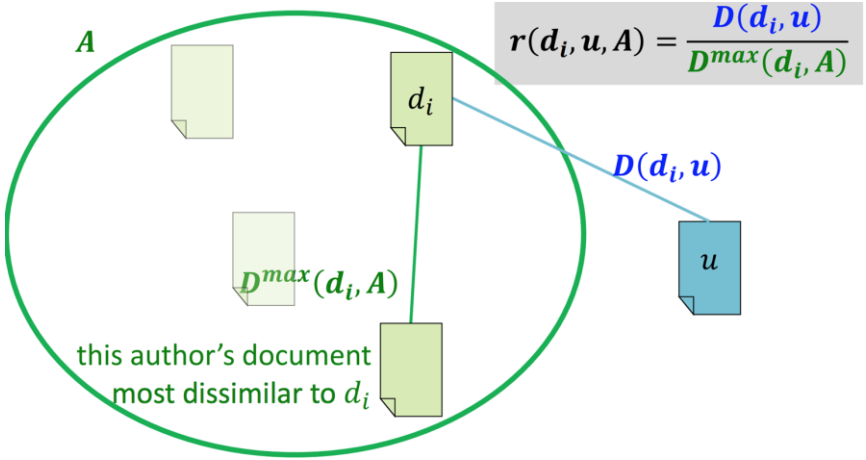
CNG dissimilarity between these documents

$$D = \sum_{x \in P_1 \cup P_2} \left(\frac{f_1(x) - f_2(x)}{\left(\frac{f_1(x) + f_2(x)}{2} \right)} \right)^2$$

Подсчет «различности»

- сравниваем профили P1 и P2
- из каждого профиля L наиболее частотных n-грамм длины n

***A** - множество документов определенного автора*



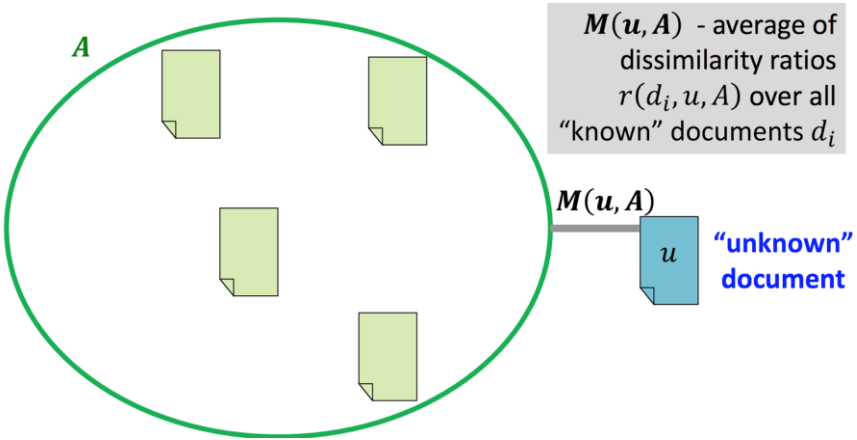
$$r(d_i, u, A) = \frac{D(d_i, u)}{D^{max}(d_i, A)}$$

D(di, u)** - отличие текущего документа **di** от тестового **u

Подсчет «различности»

- во множестве A находим документ, наиболее не похожий на **di**, то есть с наибольшим значением различности **Dmax(di, A)**;
- рассчитываем коэффициент различности **r(di, u, A)** -- насколько от текущего документа **di** отличен тестовый документ **u** в сравнении с самым непохожим на **di** документом этого автора

Подсчет «различности»



- для каждого документа множества **A** рассчитываем коэффициент различности **r** с тестовым документом **u**;
- находим **M(u, A)** -- среднее **r** по всему множеству документов данного автора **A**

Обучение

- Среднее отличие M сравнивается с параметром θ
- M <= θ -> YES M > θ -> NO
- Подбор θ с помощью логистической регрессии

```
clf = LogisticRegression(random_state=0).fit(X, y)
theta = (clf.intercept_/-clf.coef_)[0][0]
```

- Предсказание ответов с полученной θ

```
def classify(known_profile, unknown_profile, theta):
    mean_r = find_mean_ratio(known_profile, unknown_profile)
    if mean_r <= theta:
        return 'Y', mean_r
    else:
        return 'N', mean_r
```

Оценка качества

- accuracy (доля правильных ответов)

```
accuracy = accuracy_score(answers, predictions)
```

- точность подсчитывается для каждого классификатора

```
def evaluate(lang_df, test_lang_df, answers, answers_test):
    for i in range(3, 11):
        print(f'Train on char {i}-grams...')
        theta = train(lang_df, answers, i)
        predictions = predict(lang_df, theta, i)
        accuracy = accuracy_score(answers, predictions)
        print(f'Accuracy on train:\t{(accuracy*100):.2f}%')
        print(f'Theta:\t\t\t{theta:.3f}')

        predictions = predict(test_lang_df, theta, i)
        accuracy = accuracy_score(answers_test, predictions)
        print(f'Accuracy on test:\t{(accuracy*100):.2f}%')
```

Результаты

N-граммы символов

```
Train on char 3-grams...
Accuracy on train:      65.71%
Theta:                  1.187
Accuracy on test:       57.65%
Train on char 4-grams...
Accuracy on train:      77.14%
Theta:                  1.198
Accuracy on test:       54.12%
Train on char 5-grams...
Accuracy on train:      77.14%
Theta:                  1.208
Accuracy on test:       51.76%
Train on char 6-grams...
Accuracy on train:      74.29%
Theta:                  1.213
Accuracy on test:       52.94%
```

```
Train on char 7-grams...
Accuracy on train:      71.43%
Theta:                  1.216
Accuracy on test:       52.94%
Train on char 8-grams...
Accuracy on train:      71.43%
Theta:                  1.217
Accuracy on test:       52.94%
Train on char 9-grams...
Accuracy on train:      68.57%
Theta:                  1.218
Accuracy on test:       51.76%
Train on char 10-grams...
Accuracy on train:      68.57%
Theta:                  1.218
Accuracy on test:       51.76%
```

N-граммы слов

```
Train on word 1-grams...
Accuracy on train:      77.14%
Theta:                  1.225
Accuracy on test:       56.47%
Train on word 2-grams...
Accuracy on train:      68.57%
Theta:                  1.216
Accuracy on test:       50.59%
Train on word 3-grams...
Accuracy on train:      65.71%
Theta:                  1.221
Accuracy on test:       49.41%
```

Результаты

Английский

```
Train on char 5-grams...
Accuracy on train:      70.00%
Theta:                  1.068
Accuracy on test:       63.33%
```

Испанский

```
Train on char 4-grams...
Accuracy on train:      85.00%
Theta:                  1.185
Accuracy on test:       66.67%
```

Греческий

```
Train on char 3-grams...
Accuracy on train:      60.00%
Theta:                  2.205
Accuracy on test:       52.00%
```

Лучшие результаты -- греческий (самый большой датасет)
Худшие результаты -- испанский (самый маленький датасет)

Ансамбли

- Самая простая логика ансамблей: собрать все предсказания одиночных классификаторов, отобрать самые популярные варианты ответа для каждой задачи и рассчитать accuracy
- Предсказания собирались одновременно с расчётом теты и accuracy в простых классификаторах, так что никакого дополнительного обучения не потребовалось

```
Calculating ensembles on English char data...

Accuracy on test:       63.33%
Calculating ensembles on English word data...

Accuracy on test:       56.67%
```

```
Calculating ensembles on Greek char data...

Accuracy on test:       60.00%
Calculating ensembles on Greek word data...

Accuracy on test:       53.33%
```

```
Calculating ensembles on Spanish char data...

Accuracy on test:       52.00%
Calculating ensembles on Spanish word data...

Accuracy on test:       52.00%
```

Ансамбли

Проблема: если плохо работающих классификаторов будет больше, чем хорошо работающих, то их неправильные ответы перевесят и испортят общую accuracy ансамбля

Возможные решения:

- брать в ансамбль только ответы N самых лучших классификаторов
- брать ответы всех, но нормировать веса так, чтобы приоритет был у лучших

Ансамбли

“Брать в ансамбль только ответы N самых лучших классификаторов”

- Как определить N?
- Для начала N = половина от имеющихся уникальных ответов
- Далее бесконечный простор для разных способов высчитывания оптимального N

```
Calculating ensembles on English char data...

Accuracy on test:       63.33%
Calculating ensembles on English word data...

Accuracy on test:       56.67%
```

→ 63.33%

→ 63.33% + 6.66%

```
Calculating ensembles on Greek char data...

Accuracy on test:       60.00%
Calculating ensembles on Greek word data...

Accuracy on test:       53.33%
```

→ 66.67% + 6.67%

→ 63.33% + 10%

```
Calculating ensembles on Spanish char data...

Accuracy on test:       52.00%
Calculating ensembles on Spanish word data...

Accuracy on test:       52.00%
```

→

→

Невозможно подсчитать, т.к. уникальный набор ответов был всего один

Сравнение результатов

- Наша accuracy получилась ниже, чем у авторов статьи
- Ансамбли не помогли улучшить качество

```
Evaluating on Russian data...

Train on char 3-grams...
Accuracy on train:      90.00%
Theta:                  1.513
Accuracy on test:       66.67%
Train on char 4-grams...
Accuracy on train:      80.00%
Theta:                  1.542
Accuracy on test:       66.67%
Train on char 5-grams...
Accuracy on train:      70.00%
Theta:                  1.629
Accuracy on test:       50.00%
Train on char 6-grams...
Accuracy on train:      60.00%
Theta:                  1.717
Accuracy on test:       50.00%
Train on char 7-grams...
Accuracy on train:      60.00%
Theta:                  1.788
Accuracy on test:       50.00%
```

Попытки улучшить точность

- Попробовали использовать н-граммы токенов, а не слов
- Попробовали поменять местами тест и трейн оригинального датасета
- Попробовали запустить на своем (русском) датасете, не сбалансированном по времени, жанру, объему

Возможные причины

- Маленький размер выборки
- Не проводилось усечение текстов

Спасибо!