Домашнее задание №3. Spellchecker

Информационный поиск

Как было сказано на лекции, в этой домашней работе нужно разработать систему исправления опечаток. За полное задание можно получить 30 баллов. 15 баллов даются за компоненты вашей системы:

- Модель языка (2 балла)
- Модель ошибок (3 балла)
- Генератор исправлений с помощью нечеткого поиска в боре (4 балла)
- Классификатор (2 балла)
- Итерации (1 балл)
- Разные типы исправлений: словарные, split, join и раскладка (3 балла)

Остальные 15 баллов даются за качество вашей системы. По 5 баллов за качество по каждому пакету (Need no fix, Need fix, (Need split + Need join)/ 2). Четких критериев нет, поэтому эта оценка может меняться. Но примерные критерии следующие: выше 90% - 5 баллов, ниже 10% - 0 баллов.

Для создания модели языка и модели ошибок, в также для обучения классификатора можно использовать файл с запросами (скачать). В нем содержатся запросы и, через табуляцию, исправление (если есть).

Для проверки решения развернута система автоматической проверки. Чтобы ей воспользоваться, нужно сделать следующее:
1. Скрипт, исправляющий опечатки должен называться spellchecker.py
2. Он должен принимать на вход строки с запросами и для каждого возвращать его исправленный вариант. Обратите внимание, запрос может быть уже без ошибки, тогда должен вернуться оригинальный запрос.
3. Перед запуском spellchecker.py будут запускаться preinstall.sh и indexer.py. Первый скрипт позволяет вам настроить окружение. Второй можно использовать для создания модели языка и модели ошибок. В папке, где он запускается обязательно будет присутствовать файл queries_all.txt

4. Ваша система должна работать со скоростью не менее 10 запросов в секунду (если дольше, то будут сниматься баллы).
5. Все скрипты нужно обернуть в архив (tgz) и послать по адресу sfera.spellchecker@mail.ru
6. Через некоторое время вам придет ответ с указанием того, сколько правильных ответов дала ваша система. Проверяется одновременно и то, что запрос с ошибкой исправляется и то, что запрос без ошибки не исправляется. Самый простой скрипт, возвращающий введенный запрос, отработает со следующим результатом:
No need fix: 1000/1000 (100%)
Need fix: 0/1000 (0.00%)
Need join: 0/1000 (0.00%)
Need split: 0/1000 (0.00%)
Т.е. везде где не нужно исправлять, он ничего не исправил (и это хорошо), но везде, где нужно исправить, он ничего не исправил (и это плохо).
Вам нужно набрать как можно больше правильных ответов в каждой категории.
7. Письмо с итоговым результатом пишите с префиксом [FINAL]

Дедлайн: 1 мая

P.S. Если у вас остались вопросы по заданию, пишите их в комментариях к данному посту.

P.P.S: Указывайте в заголовке письма слово test, если хотите протестировать качество на небольшом объеме данных (т.к. если ваша программа работает медленно, то придется долго ждать пока пройдет полный набор тестов)

Внимание!
В письмах присылается STDERR вашего скрипта. Специально, чтобы вы могли понять, в чем ошибка. Использовать это для получения тестовых данных нечестно. Это будет караться баллами вплоть до незачета этой домашки. Тестировать свою программу вы можете на тех данных, которые у вас есть (в тесте данные аналогичные).

P.P.P.S: Понятно, что если вы пришлете скрипт, который просто выводит то, что прочитал, то у вас будет в No Need Fix 100%, а в других 0%. Очевидно, что 5 баллов за No Need Fix вы таким образом не получите. Оценка за No Need Fix не может быть выше, чем max(Need fix, Need Join, Need Split).