X

Style Change Detection

Зуева Надежда 594 группа
кафедра Анализа Данных
Факультет Инноваций и Высоких Технологий
Московский Физико-Технический Институт

курс "Моя первая научная публикация"

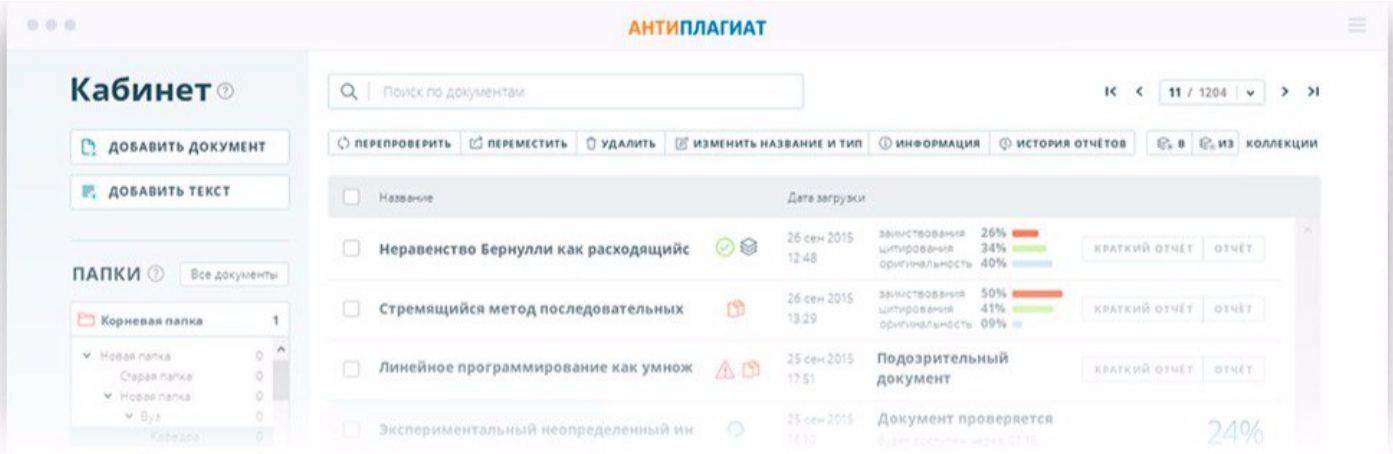
Цель исследования

Глобально: научиться как можно лучше отличать моноавторные документы от мультиавторных в случае, когда нет внешней опорной коллекции

Локально: улучшить метрику качества в соревновании PAN-2018 по сравнению с существующими методами путем использования технологии GAN

Литература

1. Stein, B., Barrón Cedeño, L.A., Eiselt, A., Potthast, M., Rosso, P.: Overview of the 3rd international competition on plagiarism detection.
2. http://pan.webis.de/clef18/pan18-web/author-identification.html
3. H. A. Chowdhury, D. K. Bhattacharyya, Plagiarism: Taxonomy, tools and detection techniques
4. https://pdfs.semanticscholar.org/1011/6d82a8438c78877a8a142be47c4ee86
5. Zechner, M., Muhr, M., Kern, R., Granitzer, M.: External and intrinsic plagiarism detection using vector space models. Proc. SEPLN. vol. 32 (2009)
6. Stamatatos, E., Tschuggnall, M., Verhoeven, B., Daelemans, W., Specht, G., Stein, B.,



АНТИПЛАГИАТ
образовательный стандарт и гарант реализации государственных решений

Система №1
на рынке поиска заимствований русскоязычных текстов и документов стран СНГ

12 лет успеха
резидент фонда «Сколково»

85% студентов
учатся в ВУЗах, использующих систему АНТИПЛАГИАТ

Постановка задачи

Пусть нам дана коллекция текстовых документов D, здесь d_k — каждый отдельный текстовый документ.
В пару к d_k ставится в соответствия $t_k \in 0, 1$. Она принимает значение 0, если документ моноавторный и 1, если авторов несколько. Требуется построить алгоритм-классификатор $a : D \to y_k$, который получает на вход документ и проверяет его на плагиат, попутно минимизируя *функцию ошибки erf*:

$$erf = -\frac{1}{|D|} \sum [t_k log y_k + (1 - t_k) log(1 - y_k)$$

$$a = \frac{1}{|D|}] \sum argmin_a erf(y_k, t_k)$$

Принципы работы

Пусть X это пространство объектов, в нашем случае это статьи из соревнования PAN-2018.
$D : X \to (0, 1)$ — функция-дискриминатор. Эта функция принимает на вход объект $x \in X$ (текст некоторого размера) и возвращает вероятность того
, что входной текст является мультиавторным. $G : Z \to X$ — функция-генератор. Она принимает значение $z \in Z$ и выдает объект пространства X.

Принцип работы

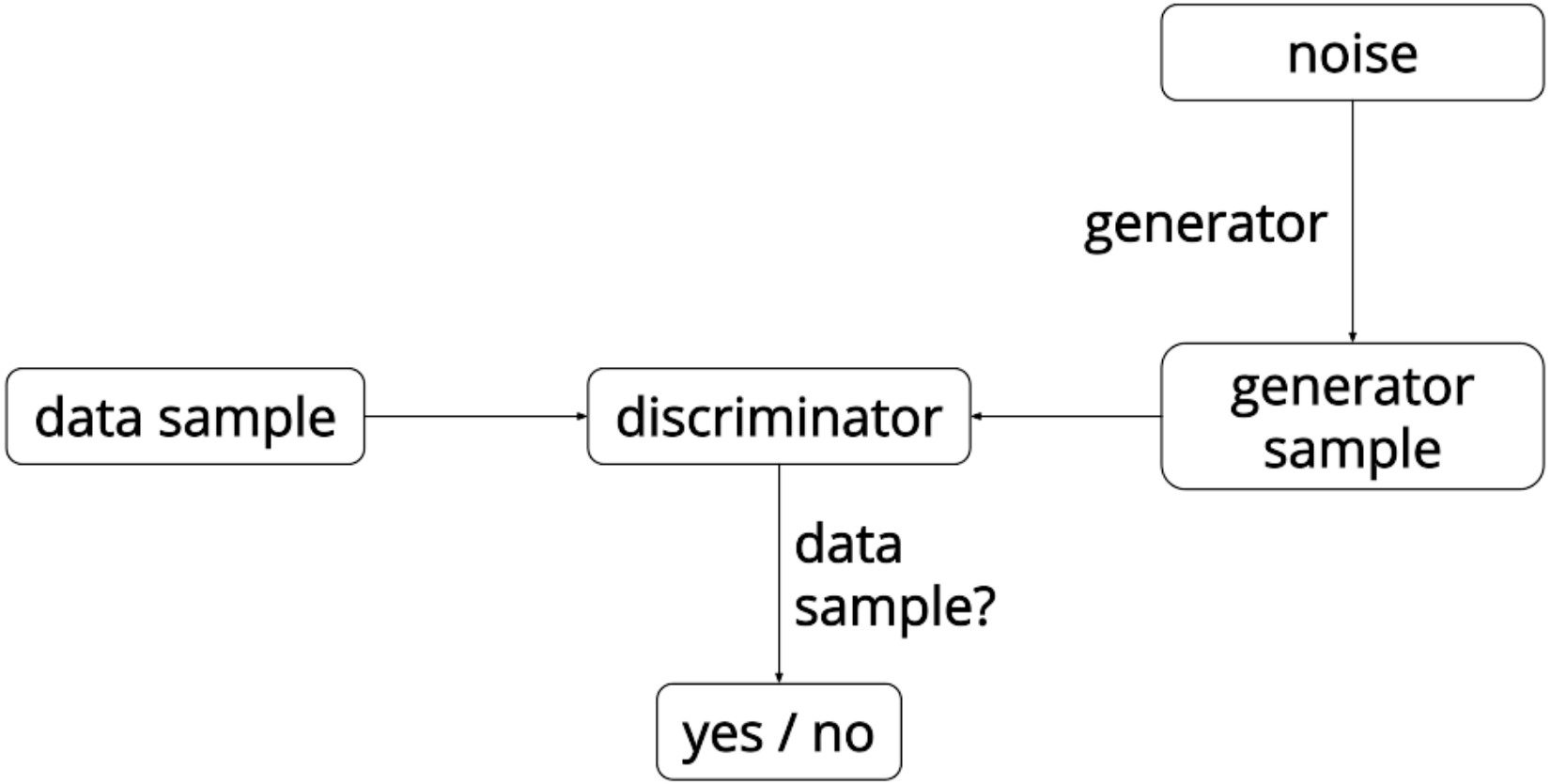


рис.1

Принцип работы

Переформулируя задачу обмана дискриминатора на вероятностном языке мы получаем, что необходимо *максимизировать* вероятность, выдаваемую идеальным дискриминатором на сгенерированных примерах. Таким образом оптимальный генератор находится как

$$G* = argmax_g E_{z\ q(x)} D_k(G(z))$$

. Известно, что $log(x)$ монотонно возрастает и не меняет положения экстремумов аргумента, то эту формулу переписать в виде:

$$G* = argmax_g E_{z\ q(x)} logD_k(G(z)),$$

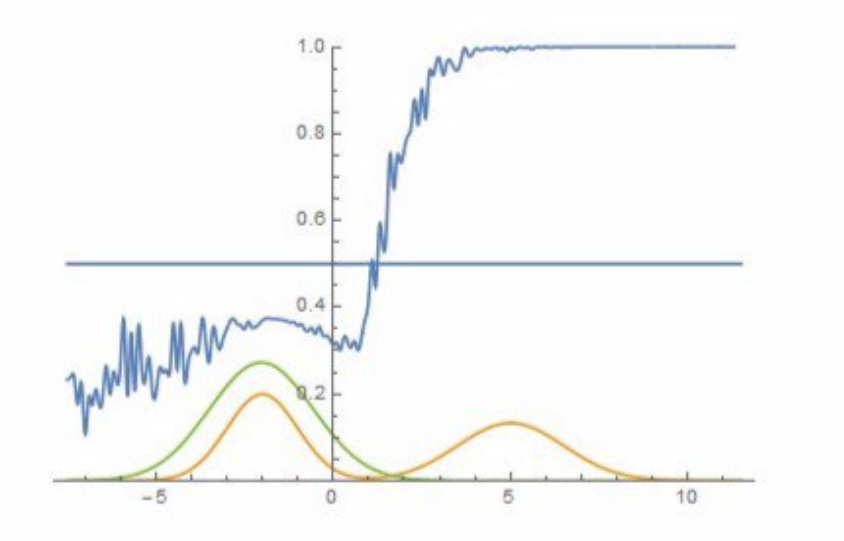
что будет удобно далее.

Принцип работы

В реальности идеального дискриминатора нет. Так как задача дискриминатора — предоставлять сигнал для обучения генератора, вместо идеального дискриминатора достаточно взять дискриминатор, *идеально отделяющий настоящие примеры от сгенерированных текущим генератором*, т.е. идеальный только на подмножестве X из которого генерируются примеры текущим генератором.

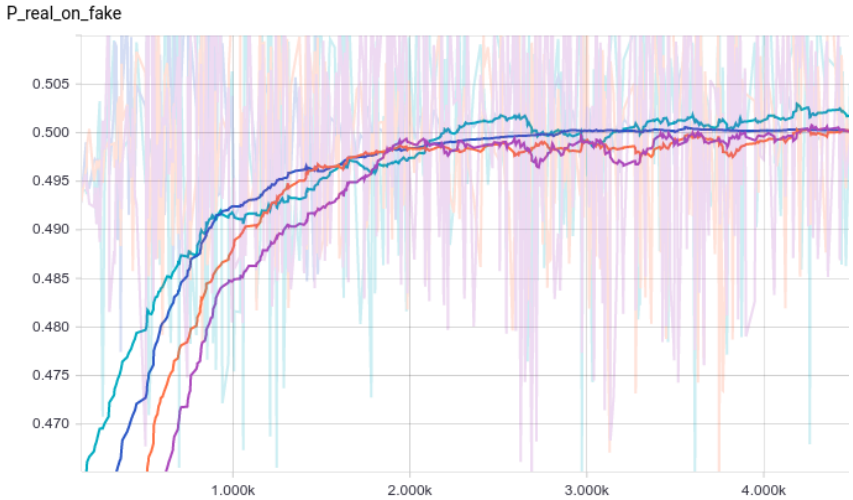
Пример

Используется датасет **PAN-2018**

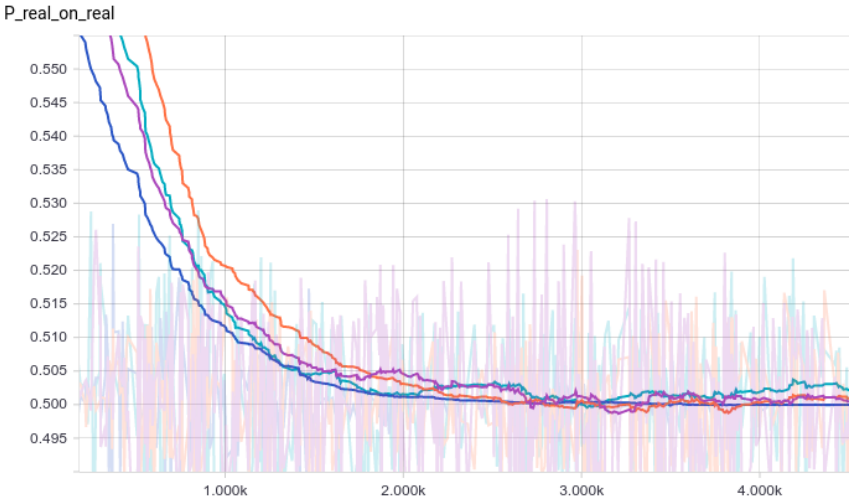


оранжевая кривая — плотность распределения реальных данных, зеленая кривая это плотность распределения генерируемых примеров, синяя кривая — результат работы дискриминатора, т.е. вероятность примера быть настоящим. dLossReal: 0.7114341 dLossFake: 0.74694636

Анализ результатов из примера



Вероятность классификации дискриминатором реального примера как реального.



Вероятность классификации дискриминатором сгенерированного

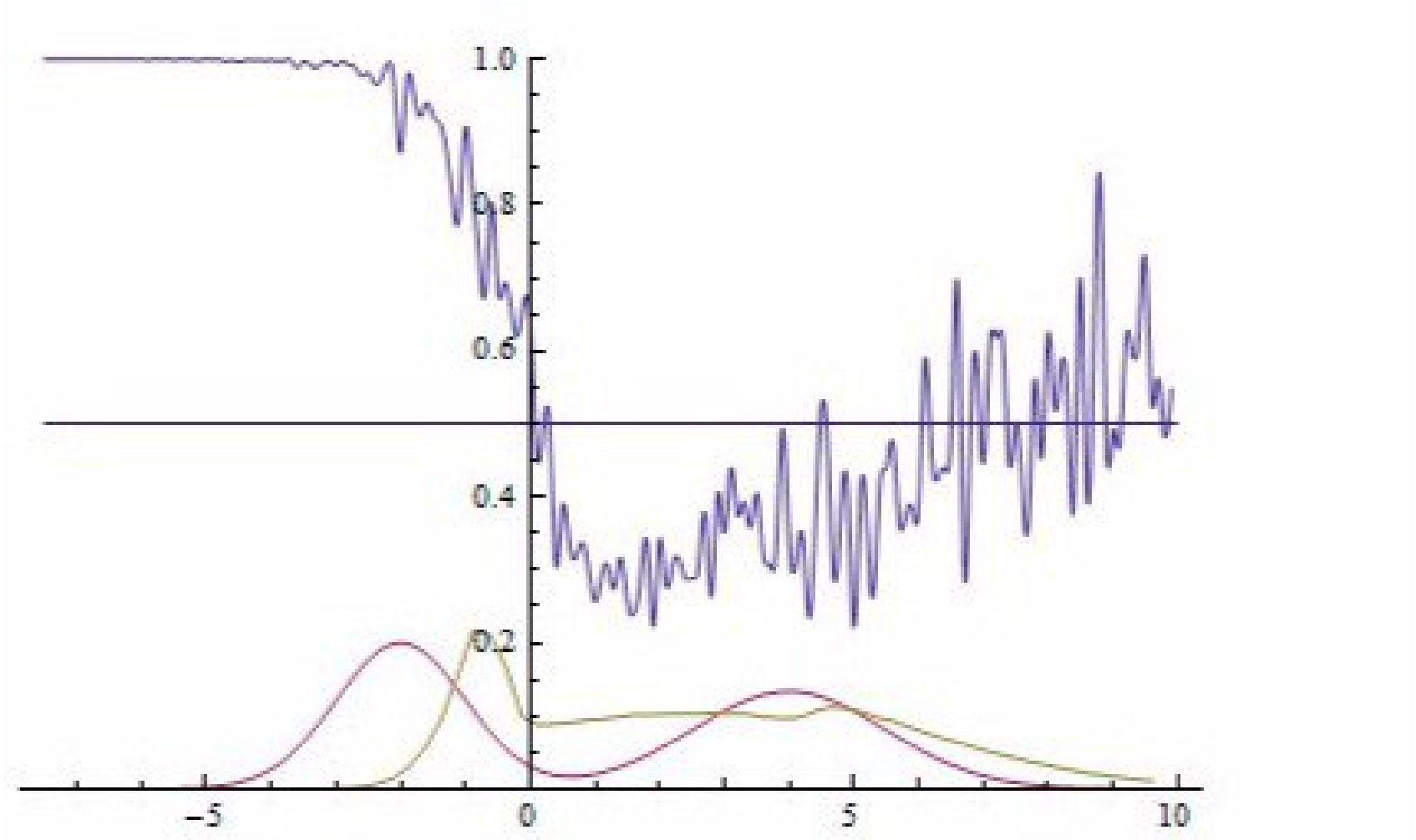
Анализ результатов

Из-за большого количества параметров обучение стало гораздо более шумным. Дискриминаторы всех моделей сходятся к одному результату, но ведут себя нестабильно вокруг этой точки. Посмотрим на форму генератора. Его форма похожа на распределение *TwinPeaks*.

Анализ результатов

Самая регуляризованная модель показала себя лучше всех. Она выучила две моды, примерно совпадающие с модами распределения данных. Размеры пиков тоже не очень точно, но приближают распределение данных. Таким образом, нейросетевой генератор способен выучить мультимодальное распределение данных.

Анализ результатов



обучение мультимодальной модели dLossReal: 0.7012983 dLossFake: 0.72894649

Вывод

Итак, если использовать более сложное устройтство генератора и обучаться на больших данных, то можно будет значительно улучшить показания на метриках качества.