Multi-Source Spatial Entity Linkage

[Основная Идея] Что такое Spatial Entity (SE)?

SE - это объект состоящий из координат и атрибутов, таких как:

- Текстовые атрибуты - название географической точки, адрес, сайт, описание.
- Семантические - категории, ключевые слова, метаданные.
- Числовые - даты, время, номер телефона и т.д.

Пример: Ресторан "Star Pizza" с координатами (56.716 10.114), описываемый ключевыми словами: "pizza, fast food" по адресу "Storegade 31".

Авторы также делают акцент на то, что в разных источниках Physical Entity (PE) (Физический объект, который мы пытаемся смэтчить), может быть описан разными SE (отличия в ключевых словах, отсутствие некоторой информации)ю

Этапы предложенного решения задачи SE Linkage

1. Spatial Blocking

   Попарно сравнивать все возможные пары долго, поэтому авторы используют модификацию QuadTree.(Дерево, которое рекурсивно бьет поверхность на квадраты, таким образом, что в листьях оказываются соседи по расстоянию)

   Их модификация учитывает плотность точек и максимальное расстояние между ними, вместо количества элементов в разбиении.

   +Пограничные точки добавляются сразу в несколько листьев, чтобы не было ситуации, когда SE одного и того же PE лежат в разных сплитах.

2. Pairwise Comparisons

   Данный этап проводится в каждом сплите.

   - Для текстовых атрибутов (кроме адреса) используется:

$$TextSim(s_1, s_2) = (1 - \frac{d(s_1, s_2)}{max(|s_1|, |s_2|)})$$

   $d(s_1, s_2)$- расстояние Левенштейна.

- Похожесть адресов имеет три значения основанные на частичном совпадении имен:

  its real semantics. In our case, we perform some data cleaning (removing commas, punctuation marks, lowercase, etc.), and then we search for equality ($s_1 = s_2$) or inclusion ($s_1 \subset s_2$ or $s_2 \subset s_1$). We assign a similarity of 1 in the case of equality and 0.9 in the case of inclusion. Otherwise, the strings are not considered the same.

- Для семантических атрибутов используется Wu&Palmer similarity measure старая мера, которая ходит по сети WordNet и считает глубины слов и их общего подмножества. Итоговая Similarity для признаков вычисляется, как максимальная схожесть по каждому из ключевых слов, категорий и т.д.
- Числовые атрибуты просто сравниваются посимвольно.

3. Labelling the Pairs

   Авторы предлагают ввести utility функцию, которая будет принимать на вход две SE. Например: сумма всех similarity для двух объектов.

   Далее итеративно происходят следующие шаги для top-k мэтчей:

   1. Вычисление utility для всех пар SE в листе модификации Quad Tree.
   2. Пара с максимальной utility отмечается как мэтч.
   3. Пара с максимальной utility выкидывается из листа.
   4. Снова выполняется пункт 1, если количество мэтчей меньше k.

Результаты

Авторы репортят 0.60 Precision и 0.87 Recall на датасете из 777K пар. В сравнении с другими решениями они получили лучшее значение F1- меры.

Репозиторий: Нет

Датасет: Нет

Можно ли использовать в нашем проекте?

Авторы описали алгоритм свою модификацию QuadTree, которую, наверное, можно было бы использовать у нас, чтобы отсеять часть потенциальных FP пар.

Profile Reconciliation Through Dynamic Activities Across Social Networks

[Основная Идея] Использование временного окна для мэтчинга.

Два профиля, скорее всего, принадлежат одному и тому же человеку, если они содержат похожие посты в разных сетях, которые запостили в одном небольшом временном окне.

Как представлен аккаунт?

- Гео-данные (гео-метки с фотографий и постов).
- Фото.
- Текст поста.
- Теги.
- Username.
- Настоящее имя.

Для каждого из признаков считается свой similarity score

- Для гео-данных происходит обращение в апишку Google Maps, откуда вытаскивается время кратчайшего способа передвижения.
- Для Фото используют OpenCV, чтобы получить вектор признаков, на котором считают cosine-similarity.
- Текст векторизуют с помощью tf-idf и опять считают cosine-similarity.
- Теги обрабатываются так же как и текст, но вынесены отдельно, чтобы подчеркнуть их важность.
- Для определения похожести username считается (1 - расстояние Левенштейна).
- Настоящее имя бьется на токены и по спискам токенов считается коэффициент Жаккара.

Как работает алгоритм?

Все посты пользователя разбиваются на несколько временных окон и в каждом окне считается вектор similarity для каждой пары постов. На этих векторах учится логистическая регрессия. Если для хотя бы одной пары постов логистическая регрессия предсказала мэтч, то алгоритм говорит, что представленные пользователи - это один и тот же человек.

Для того чтобы ускорить процесс, авторы отсекают пары постов, если время кратчайшего способа передвижения между локациями постов больше временного окна постов. (Т.е. автор поста не смог бы доехать до точки, из которой был сделан пост)

Результаты

Approach	Recall	Precision	AUC
Edwards et al.	0.407	0.970	0.788
Gianluca et al. - 1 rule	0.425	1.000	0.635
Gianluca et al. - 2 rule	0.423	0.995	0.711
Pachenko et al.	0.222	0.990	0.611
Bubble matching 20 min	0.631	0.822	0.815
Bubble matching 1 h	0.698	0.886	0.849

Авторы добились достаточно неплохих метрик (оранжевым подсвечен размер временного окна), однако проверяли они алгоритм всего на 1.3К пар.

Репозиторий: Нет

Датасет: Нет

Можно ли использовать в нашем проекте?

Идея их алгоритма и отсечки неподходящих пар достаточно хорошо переносится на нашу задачу.

Положим, что у нас есть два красноармейца в разных деревьях, если это один и тот же человек, то они имели один и тот же боевой путь. Также положим, что один пользователь хорошо заполнил свое дерево, а другой пользователь вбил только основные битвы. Подобный алгоритм позволяет смэтчить их деревья.

Использование предложенной ими отсечки позволит убрать из потенциальных мэтчей полных тёзок. Например: обоих зовут Иван Иванов, но один призывался из одного военкомата, а другой из другого.

SocialMatching++: A Novel Approach for Interlinking User Profiles on Social Networks

[Основная Идея] Что такое Life Event?

Life Event - это редкое событие, такое как свадьба, выпускной и т.д.

Идея авторов статьи заключается в том, что даже если пользователь не активно ведет одну или несколько соц. сетей уж о Life Event он должен сообщить в своих постах.

Как представлен аккаунт?

- Life Event
- Bio (Краткая информация о себе, которую составил пользователь)

Как работает алгоритм?

Авторы используют простенький NER, чтобы вытащить сущности из bio потенциальных кандидатов. Если сущности одинаковые, то алгоритм выдает мэтч.

Если же по bio классифицировать не удалось, из постов в одной соц. сети вытаскиваются Life Events и, с некоторым окном, ищут Life Events за ту же дату в другой соц. сети и считают между ними cos-sim.

У них очень странно написана формула, но есть репозиторий, поэтому это не так критично.

Результаты

Они репортят 0.8 - 0.9 Precision, однако авторы собрали 1000 сэмплов, а метрику считают только на 50 из них. Выглядит странно.

Репозиторий: Есть

Датасет: Нет

Можно ли использовать в нашем проекте?

Скорее всего, да, их идея в принципе не особо привязана к соц. сетям. Подобные Life Events в качестве признаков могут позволить лучше различать людей в генеалогических деревьях.

Имеются ввиду еще более редкие события (такие как, участие в военных конфликтах или прочие заслуги, про которые говорят в семье), чем те, о которые имеют ввиду авторы, ведь генеалогическое дерева это по сути список Life Events

Tuser3 : A Profile Matching Based Algorithm Across Three Heterogeneous Social Networks

Как представлен аккаунт?

- Имя
- Фото
- Bio
- UGC (Контент созданный пользователем)

Для каждого из признаков считается свой similarity score

- Для имени с помощью Модифицированного алгоритма Бойера-Мура
- Для фото используется microsoft Face API (Бинарный классификатор мэтч/не мэтч)
- Для bio - cos-sim от векторизованных текстов

Из UGC выделяется еще один набор признаков:

- ВременнЫе - частота постов одного аккаунта вычитается из частоты постов другого аккаунта.
- Эмоциональные - Используется несколько апишек, чтобы вытащить похожесть признаков эмоционального окраса текста.
- Стиль письма - Авторы используют статью, где был предложен набор частотных признаков для определения стиля письма. Similarity считается также, как и в случае временных признаков.
- n-граммы - Считается нормированное пересечение n-грамм.

$$Sim(n-grams)_{(P_S,P_T)} = \frac{|n-grams(P_S) \cap n-grams(P_T)|}{Min(|n-grams(P_S)|, |n-grams(P_T)|)}$$

Как работает алгоритм?

- Чтобы не считать similarity по всем возможным парам они разбивают имена по пробелам и на части, и ищут все перестановки подобных токенов в другой сети.
- После чего считается средняя Similarity по всем атрибутам и на основе порога делается вывод мэтч/не мэтч

$$Score(P_S, P_T) = \frac{\sum Sim(Attribute)_{(P_S,P_T)}}{\sum Attribute}$$

- После этого авторы тренируют простые алгоритмы обучения с учителем, чтобы среди мэтчей выбрать один верный.

Результаты

	Match-name		Tuser2		Tuser3	
	MAP	AUC	MAP	AUC	MAP	AUC
TW-TM	0.32	0.23	0.4319	0.5021	**0.4905**	**0.5597**
TW-YT	0.4708	0.3768	0.3587	0.4317	**0.4996**	**0.5306**
TW-TW	0.2982	0.2512	0.2637	0.4654	**0.4369**	**0.5046**

Алгоритм показывает себя чуть лучше случайного классификатора.

Авторы также репортят какие-то заоблачные цифры Precision/Recall/F1 для своей supervised matching модельки.

		Match-name			Tuser2			Tuser3		
		Precision	Recall	F-Measure	Precision	Recall	F-Measure	Precision	Recall	F-Measure
KNN	TW-TM	35.7	33.54	34.06	80	50	66.7	**100**	**96.9**	**98.4**
	TW-YT	38.4	37.22	37.8	90.1	66.7	85	**95.33**	**82.9**	**90.6**
	TW-TW	35.7	33.54	34.06	80	50	66.7	**100**	**96.9**	**98.4**
Bayes Net	TW-TM	39.2	32	33.75	100	99.5	99.7	**100**	**99.4**	**99.7**
	TW-YT	38.4	35.61	37.1	100	100	100	**100**	**100**	**100**
	TW-TW	30.2	19.7	24.71	83.33	55.5	71.4	**100**	**95.8**	**97.9**
Random forest	TW-TM	38.4	30.22	30.8	80	66.7	70.2	**100**	**100**	**100**
	TW-YT	38.4	30.22	30.8	80	66.7	70.2	**100**	**100**	**100**
	TW-TW	39.66	31.1	32.7	90.4	79.2	88.4	**96.33**	**89.9**	**94.7**
SMO	TW-TM	39.2	24	27.4	93.4	82.8	90.6	**96.66**	**87.9**	**93.6**
	TW-YT	38.4	30.22	30.8	96.33	83.3	94.4	**100**	**93.6**	**96.7**
	TW-TW	39.66	33.8	34.2	100	97.8	98.9	**100**	**100**	**100**
Adaboost	TW-TM	39.2	25.33	26.6	100	92.4	96	**100**	**92.4**	**96**
	TW-YT	38.4	37.22	37.8	100	83.3	97.8	**100**	**96.4**	**98.2**
	TW-TW	35.7	28.3	31.1	73.33	41.7	58.8	**96.66**	**81.9**	**90.1**

Репозиторий: Нет

Датасет: Нет

Можно ли использовать в нашем проекте?

Нет, нельзя перенести что-то из их статьи в наш проект. Также полученные метрики вызывают вопросы.