“Извлечение информации с помощью графовых нейронных сетей”

Войтецкий Артем

1. Введение

В современном мире большую часть информации люди берут из интернета с различных сайтов, Однако, часто страницы содержат большое количество информации, которая может быть несущественной для пользователя. Это может приводить к негативным последствиям, таким как утомительность при чтении, отвлечение от главного содержания и потеря времени на поиск нужной информации.

Для того чтобы избежать этих проблем, важно иметь систему, которая способна определять важные блоки на веб-странице для пользователя. Такая система может автоматически выделять основное содержание, исключая все лишнее. Это поможет пользователям быстрее получать нужную информацию, а также позволит упростить использование текстовых классификаторов и суммарайзеров, так как на вход им можно будет подавать только важные блоки, что увеличит общую эффективность.

В данном проекте будет представлен подход к решению данной задачи с помощью графовых нейронных сетей различных типов и конфигураций, а также будут проанализированы результаты его применения на реальных сайтах.

2. Цель работы

Целью данной работы является разработка графовой нейронной сети, способной определять важные части сайта, используя различные представления веб страницы в виде графа.

3. Данные

В качестве входных данных были использованы результаты работы скрипта, который извлекает все элементы html страницы и их параметры. У каждого такого элемента имеются данные о его местоположении на странице и занимаемой площади, цвете и размере шрифта, цвете фона блока, имеется ли в нем изображение и длина текста в блоке.

4. Графы

Из полученных данных убирались все блоки со скриптами, ссылками и стилями, так как они не представляют интереса для данной задачи. Также были убраны все блоки, которые имеют нулевую площадь или находятся за границами отображаемой области.

Отобранные данные преобразовывались в два вида графов: основанные на DOM-дереве, основанные на данных о соседстве двух блоков. Параметры вершин у графов были идентичны и имели по 9 признаков: позиция блока на странице (2 признака), его размер (2 признака), его заполненность текстом (1 признак), контраст текста и фона (3 признака) и наличие в блоке картинки (1 признак).

В итоге получилось по 1824 графа каждого вида, которые в среднем имеют такие параметры:

	DOM-дерево	по признаку соседства
среднее количество вершин	775	785
среднее количество ребер	774	13104
среднее количество полезных блоков	48,5	47

Как видно из этой статистики, среднее количество ребер в графе, основанном на DOM-дереве значительно меньше. Дальнейшее исследование должно выявить, помогают ли дополнительные ребра в графе классифицировать элемент.

5. Графовые нейронные сети

В этой работе были использованы графовые нейронные сети 4 различных видов: GCNConv, GATConv, GATv2Conv, SAGEConv. Все они были написаны на языке python с помощью библиотеки pytorch geometric.

Graph Convolutional Network – это сверточная нейронная сеть на графе, но вместо ядра свертки используются процессы агрегации и обновления. Первая операция агрегирует данные признаков соседей и текущей вершины. Вторая операция выполняет кодирование данной вершины на основе результата агрегации. В GCNConv используется поэлементное среднее в качестве агрегирующей функции, а затем берется взвешенная сумма.

Graph Attention Network отличается тем, что агрегация признаков происходит с использованием механизма внимания. Он представляет из себя скрытые слои, которые присваивают соседям узла индивидуальные веса, которые учитываются при агрегировании. Разница между двумя подтипами заключается в том, что GATConv использует статический механизм внимания, а GATv2Conv - динамический.

SAGEConv – это еще один вариант нейронной сети, который отличается методом агрегации. Такая нейронная сеть позволяет использовать в качестве функций агрегации среднее, pooling или LSTM. В качестве операции обновления выступает конкатенация. Преимуществом SAGEConv является способность обучаться на небольшом количестве данных.

6. Тренировка и тестирование моделей

Данные были разбиты на две категории: тренировочные и тестовые, в отношении 75%:25%.

В ходе работы были исследованы нейронные сети с различным числом слоев. Лучше всего себя показали конфигурации с 1 и 2 слоями. Для многослойных нейронных сетей также подбирались промежуточные параметры. Также было исследовано влияние количества эпох тренировки на модели.

Всего было обучено и протестировано не меньше 100 моделей каждой конфигурации. В процессе исследования был сделан вывод, что модели достаточно быстро переобучаются, так как достигать лучших показателей на тестовой выборке получалось тогда, когда тренировочных эпох было достаточно мало.

7. Результаты

Наиболее интересные из полученных результатов представлены в таблице ниже:

Модель	DOM-дерево		по признаку соседства	
	F1	ROC-AUC	F1	ROC-AUC
GCNConv 1 слой	0.514	0.834	0.518	0.798
GCNConv 2 слоя	0.454	0.814	0.521	0.794
GATConv 1 слой	0.61	0.855	0.417	0.724
GATConv 2 слоя	0.642	0.86	0.369	0.72

GATv2Conv 1 слой	0.617	0.808	0.462	0.792
GATv2Conv 2 слоя	**0.645**	**0.893**	0.39	0.767
SAGEConv 1 слой	0.625	0.792	**0.635**	**0.909**
SAGEConv 2 слоя	**0.647**	**0.824**	**0.656**	**0.876**

Как видно из таблицы с результатами, только для нейронный сетей типа GAT имеется значительное различие в типе графа, которое строится из исходных данных.

От смены типа графа GATv2 сети смогли конкурировать с SAGE, которые доминировали при другом построении графа.

SAGE же немного проиграл в метриках при переходе от графа по признаку соседства к DOM-дереву. Также стоит отметить, что только этот тип нейронных сетей смог себя достойно показать при построении графа по признаку соседства.

8. Выводы

Исходя из результатов работы можно сделать вывод, что переход от DOM-дерева к графу, основанному на соседстве блоков добавляет множество новых ребер, но далеко не всегда это хорошо работает. Также это заметно увеличивает время, необходимое на обучение моделей. Примерно в 1.5-2 раза, а метрики F1 отличаются на уровне погрешности.

Также я предполагаю, что GAT сети не смогли показать достойный результат во втором способе построения графа потому, что механизм внимания должен определить наиболее важных соседей, но, как видно из статистики графов из пункта 4, соседей получается

слишком много, из-за чего главная особенность этой нейронной сети не может начать работать или ей нужно слишком много времени на обучение.

GCNConv в свою очередь оказался малочувствителен при переходе от одного графа к другому. Тут я делаю предположение, что просто недостаточно тестировал сеть с 2 слоями на DOM-дереве и результаты там бы были аналогичные.

Переход от одного слоя нейронной сети к двум зачастую давал прирост качества модели. Здесь я не беру в рассмотрение GAT сети на графе, основанном на соседстве, так как основная причина их плохих результатов заключается не в количестве слоев модели, а в самом графе.

9. Дальнейшая работа

Я вижу потенциал графовых нейронных сетей для задачи определения важных блоков на странице, а поэтому продолжу заниматься совершенствованием нейронных сетей. Я планирую найти способы уменьшить графы сайтов, чтобы хранить как можно меньше промежуточных вершин, а также уменьшить количество ребер в графе, основанном на соседстве.

10. Ссылка на проект

https://github.com/MrARVO/Information-Extraction