Цель работы

Разработать приложение для Apache Spark в автономном режиме. Проанализировать литературное произведение (Алексей Николаевич Толстой – Эмигранты) с помощью Spark RDD.

Краткая теория

Apache Spark — фреймворк с открытым исходным кодом для реализации распределённой обработки неструктурированных и слабоструктурированных данных, входящий в экосистему проектов Hadoop. В отличие от классического обработчика из ядра Hadoop, реализующего двухуровневую концепцию MapReduce с хранением промежуточных данных на накопителях. Данный подход (MapReduce) используется в основном, для распределенных вычислений (на нескольких кластерах) больших объемов данных. Сначала выполняется шаг Map, то есть информация предварительно обрабатывается. На шаге Reduce происходит выполнение определенных функций высшего порядка, приводящих к конечному результату. Spark работает в парадигме резидентных вычислений — обрабатывает данные в оперативной памяти, благодаря чему позволяет получать значительный выигрыш в скорости работы для некоторых классов задач, в частности, возможность многократного доступа к загруженным в память пользовательским данным делает библиотеку привлекательной для алгоритмов машинного обучения.

Структуры данных RDD

RDD – это распределенная коллекция данных, расположенных по нескольким узлам кластера, набор объектов Java или Scala, представляющих данные. RDD работает со структурированными и с неструктурированные данными. Также, как DataFrame и DataSet, RDD не выводит схему загруженных данных и требует от пользователя ее указания.

RDD-коллекция сериализуется каждый раз, когда Spark требуется распределить данные внутри кластера или записать информацию на диск. Затраты на сериализацию отдельных объектов Java и Scala являются дорогостоящими, т.к. выполняется отправка данных и структур между узлами.

Ход работы

В начале работы происходит инициализация конфигурации состояния Spark, что позволяет в дальнейшем подключить контекст работы для прочтения файла с литературным произведением и дальнейшим взаимодействием с ним.

Для анализа я подготавливаю исходный текст с помощью устранения пунктуационных знаков: точка, запятая и так далее. В дополнение, слова данного произведения переводятся в нижний регистр для увеличения числа показателей значений.

В процессе обработки текста участвует специальный список стоп-слов, хранящийся в файле stop_words.txt. При сравнении исходного текста произведения со словами, которые представлены в этом файле, происходит устранение союзов, предлогов, глаголов и других лишних слов.

Затем происходит поиск 50 самых популярных и 50 самых редко используемых слов. Так, например, сочетание имени и фамилии Хаджет Лаше упоминается в тексте чаще всего, хотя главным персонажем он не является. Так же по наиболее часто встречающимся словам, можно понять, что местом действия является Петроград в революционную либо постреволюционную эпоху.

В процессе лабораторной работы был произведён поиск для 50 самых популярных и редко встречающихся слов, приведу пример выполняемого кода:

Top50 most common words:

(лаше,294)
(хаджет,192)
(бистрем,167)
(левант,120)
(вера,110)
(глаза,102)
(налымов,102)
(леви,89)
(юрьевна,82)
(лили,81)
(господа,74)
(россии,70)
(лисовский,62)
(левицкий,59)
(ардашев,57)
(мари,54)
(деньги,54)
(николай,53)
(генерал,49)
(глазами,49)
(бистрема,42)
(ответил,41)
(лиги,41)
(большевиков,40)
(черт,39)
(денисов,37)
(манташев,36)
(париж,36)
(василий,36)
(александр,35)
(друг,35)
(алексеевич,35)
(баль,35)
(петрович,35)
(революция,35)
(революции,34)
(большевики,34)
(голос,34)
(войны,33)
(петроград,33)
(должен,33)
(завтра,32)
(франков,32)
(русские,31)
(полиции,31)
(извольский,31)
(русских,31)
(дом,31)
(лица,31)
(граф,31)

Top50 least common words:

(графин,1)
(рассечена,1)
(эксцессы,1)
(сенсацией,1)
(списку,1)
(сознавали,1)
(бесцветной,1)
(собралось,1)
(гравюры,1)
(слепо,1)
(мучают,1)
(матросню,1)
(монархию,1)
(запахнул,1)
(свининой,1)
(превратностей,1)
(сбежали,1)
(тысячелетней,1)
(числите,1)
(освещает,1)
(вежлив,1)
(родству,1)
(поставим,1)
(зараза,1)
(обыщите,1)
(скатертью,1)
(нетрезвом,1)
(большей,1)
(польский,1)
(унылый,1)
(приобретены,1)
(зажал,1)
(кашалоты,1)
(полутораста,1)
(бульдога,1)
(ошень,1)
(желтеющей,1)
(раскаюсь,1)
(седоватые,1)
(понятнее,1)
(праву,1)
(сформированное,1)
(плечики,1)
(петербургского,1)
(разменивался,1)
(варвара,1)
(дрожащие,1)
(доставили,1)
(сдвинутыми,1)
(применения,1)

Для семантического поиска однокоренных слов используется stemmer, который позволяет выделять общей корень у родственных (однокоренных) слов. Его использование характеризовано вызов метода mapPartitions, который позволяет преобразовать каждый раздел исходного RDD в результат из нескольких элементов. Стемминг позволил внести некую точность в оценивании частоты использования слов. Так, например, многие имена собственные, либо топонимы, заметно опустились вниз в списке, так как в основном используются в одной и той де форме, в то время как обще разговорные слова имеют множество вариаций. Неожиданно много вариаций различных форм одного и того же слова, могут говорить многообразии устной речи. К примеру, слово «Большевиков» обнаружено в тексте 40 раз. В то время как различные его формы насчитывают около 118 упоминаний. Но стоит также отметить наличие ошибок и неточностей. В ряде случаев, к корню «лев» приписывается имя Леви, а к имени Лили приписываются слова лить, лился.

В процессе лабораторной работы был произведён поиск для 50 самых популярных и редко встречающихся слов с использованием стемминга.

Приведу пример выполняемого кода:

Top50 most common stems:
((лаш,List(лаше)),294)
((вер,List(верой, веря, верят, вере, верил, веру, веры, верит, верить, верю, вера, верим, веришь)),206)
((глаз,List(глазу, глазах, глаза, глазам, глазами, глазом, глаз)),196)
((хаджет,List(хаджетом, хаджет)),193)
((бистр,List(бистрему, бистрем)),183)
((левант,List(леванте, леванту, левантом, леванта, левант)),170)
((русск,List(русские, русскую, русских, русское, русского, русский, русскому, русской, русская, русскими, русском, русским)),154)
((юрьевн,List(юрьевна, юрьевну, юрьевне, юрьевной, юрьевны)),140)
((большевик,List(большевиках, большевику, большевики, большевиков, большевика, большевик, большевиком, большевиками, большевикам)),118)
((росс,List(россию, россии, россией, россия)),116)
((налым,List(налымов)),102)
((лев,List(леви, левый, левее, левому, левой, левую, левая)),100)
((дом,List(домами, домам, дома, дом, домом, домов, домах, домой, дому, доме)),98)
((лиц,List(лицах, лице, лицу, лиц, лица, лицам, лицом, лицами)),97)
((париж,List(париже, парижу, парижем, париж, парижа)),97)
((стол,List(стола, столь, стол, столов, столом, столам, столу, столы, столе, столах)),96)
((друг,List(другими, друга, другом, друг, другим, другому, другу, другую, другого)),92)
((бел,List(белую, белыми, белого, белья, белая, белье, бельем, белым, белой, белому, белое, белый, белых, белом, белые)),92)
((левицк,List(левицким, левицком, левицкий, левицкому, левицкого)),91)
((лисовск,List(лисовскому, лисовский, лисовского, лисовским)),87)
((петроград,List(петрограду, петрограде, петроградом, петрограда, петроград)),85)
((мо,List(мою, моими, моей, моего, моим, моему, мое, моемся, моих)),84)
((подня,List(поднявшись, подняло, поднял, поднялся, подняли, подняла, поднялись, поднялась, поднять, поднялось, подняться, подняв)),83)
((революц,List(революцией, революции, революцию, революций, революция)),82)
((лил,List(лился, лили)),82)
((господ,List(господа, господи, господ, господам, господь)),81)
((рук,List(руках, руке, рукой, рука, рукам, рук, руками)),80)
((лиг,List(лига, лиги, лиге, лигу, лигах, лигой)),79)
((голос,List(голосе, голосу, голосам, голосом, голосами, голоса, голос, голосов)),79)
((дел,List(делам, делом, дела, делами, делят, делах, делая, делу, делось, деле)),75)
((генера,List(генерал, генерала)),72)
((голов,List(голова, головами, головой, головы, голове)),69)

((войн,List(война, войн, войны, войной, войне, войну)),68)
((дорог,List(дорогой, дорогого, дорогую, дорогая, дорогих, дороге, дорогу, дорогие, дорог, дороги, дорога, дорогими)),68)
((ответ,List(ответил, ответило, ответили, ответила, ответим, ответ, ответа, ответить)),68)
((черн,List(черная, черному, черного, черными, черную, черной, черное, черным, черный, черных, черные, черном)),66)
((стокгольм,List(стокгольму, стокгольме, стокгольма, стокгольмом, стокгольм)),66)
((черт,List(черта, черт, чертами, черты, чертах, черте, чертям, черту, чертой)),65)
((ног,List(ноги, ногами, ног, нога, ногам, ногах, ногой, ногу)),65)
((газет,List(газету, газете, газетой, газеты, газетах, газет, газетами, газета)),65)
((нос,List(нос, носом, носит, носа, носить, носы, носил, носу, носят)),64)
((женщин,List(женщинами, женщин, женщина, женщинам, женщину, женщиной, женщинах, женщине, женщины)),64)
((час,List(часов, часами, час, часа, часам, часу, часах, часы)),64)
((деньг,List(деньгах, деньги, деньгами, деньгам)),64)
((больш,List(большей, большее, большего, большим, большими, большой, большею, большие, большая, больших, большое, большого, большую)),63)
((город,List(городе, городу, города, город, городами, городом, городам, городов)),63)
((слуша,List(слушает, слушала, слушай, слушал, слушали, слушают, слушаю, слушающих, слушать, слушайте)),62)
((сторон,List(сторонам, сторона, стороны, сторону, стороне)),61)
((юденич,List(юденичу, юденичем, юденич, юденичей, юденича)),61)
((налымов,List(налымову, налымовым, налымова)),60)

Top50 least common stems:
((поцелу,List(поцелуя)),1)
((запахнул,List(запахнул)),1)
((авансирова,List(авансировал)),1)
((маслиц,List(маслица)),1)
((сумел,List(сумели)),1)
((выбрасыв,List(выбрасывая)),1)
((по-стариковск,List(по-стариковски)),1)
((наступ,List(наступившей)),1)
((незакурен,List(незакуренную)),1)
((возраста,List(возраставшей)),1)
((что-либ,List(что-либо)),1)
((участву,List(участвуя)),1)
((самообладан,List(самообладание)),1)
((оцен,List(оценил)),1)
((находк,List(находка)),1)
((придушен,List(придушенным)),1)
((навис,List(навис)),1)
((шампан,List(шампани)),1)
((помещич,List(помещичьей)),1)
((магическ,List(магический)),1)
((бекхольм,List(бекхольм)),1)
((обруш,List(обрушился)),1)
((расстоян,List(расстоянии)),1)
((еврейчик,List(еврейчиков)),1)
((сталкива,List(сталкивали)),1)
((наглост,List(наглости)),1)
((мадрид,List(мадриде)),1)
((заступ,List(заступы)),1)
((непрактич,List(непрактичен)),1)
((невозмутим,List(невозмутимым)),1)
((пиявк,List(пиявки)),1)
((замаха,List(замахали)),1)
((благоухан,List(благоухание)),1)
((выберут,List(выберут)),1)
((выжидан,List(выжидание)),1)
((дерзк,List(дерзким)),1)
((вымога,List(вымогали)),1)
((переросш,List(переросший)),1)
((равнодуш,List(равнодушия)),1)
((вышиб,List(вышиби)),1)
((стрем,List(стремилось)),1)
((выним,List(вынимая)),1)
((закажеш,List(закажешь)),1)
((ливн,List(ливни)),1)
((кинжал,List(кинжале)),1)
((поз,List(позах)),1)
((верч,List(верчусь)),1)
((обидн,List(обидно)),1)
((киевск,List(киевская)),1)
((жалост,List(жалости)),1)

Вывод

В ходе лабораторной работы я настроил сборку проекта с помощью Maven для версии языка Scala 2.12.6. Подключил основную библиотеку для фреймворка Spark и написал исходный код программы для анализа романа А. Н. Толстого «Эмигранты». В процессе были выявлены 50 самых популярных и 50 самых редко используемых слов. Полученные результаты были дополнены процессом стемминга, что позволило глубже изучить семантический анализ текста. Благодаря расширяемому списку стоп-слов проект может быть расширен в рамках более детального рассмотрения.