Group 2

CROWD SOURCED GROCERY SHOPPING

Utkarsha Ganla, Aditya Srinivasan, Tushar Kulkarni

Problem

We are trying to achieve a crowd-sourced data analytic application that scans product price tags uploaded as an image by users and perform price analysis on the data. The results are published back to users who have set certain level of subscriptions. The main areas of interest in this application are Optical Character Recognition(OCR), Data Frameworks with Pub/ Sub support. We summarize a few sets of papers on each category mentioned above in chronological order.

Optical Character Recognition (OCR)

A Survey on Optical Character Recognition System by Noman Islam, et. al introduces an application that converts printed or handwritten text using images or other mediums into vector digitized form. OCR is a piece of software required for machines because standalone machines are not intelligent enough to understand the text from images. OCR is a combination of various Computer Science disciplines including NLP, Image Processing, Pattern Classification, etc. Based on the type of input, the OCR systems can be categorized as handwriting recognition and machine printed character recognition. The former is relatively simpler problem because characters are usually of uniform dimensions, and the positions of characters on the page can be predicted. Handwriting character recognition is a very tough job due to different writing style of user as well as different pen movements by the user for the same character. Our system will be using OCR for printed text that is recognizing text from images of price labels.[1]

OCR is a composite activity comprises different phases. First, an external source captures an image of text labels. Then various type of preprocessing is performed to improve the quality of image. Different types of filters such as averaging, min and max filters can be applied. Alternatively, different morphological operations such as erosion, dilation, opening and closing can be performed. The characters in the image are separated such that they can be passed to recognition engine. Among the simplest techniques are connected component analysis and projection profiles can be used. The segmented characters are then processes to extract different features. Based on these features, the characters are recognized. Character classification maps the features of segmented image to different categories or classes. There are different types of character classification techniques. Structural classification techniques are based on features extracted from the structure of image and uses different decision rules to classify characters. After classification, the results are not 100% correct, especially for complex languages. Post processing techniques can be performed to improve the accuracy of OCR systems. These techniques utilize natural language processing, geometric and linguistic context to correct errors in OCR results. This includes spell checking and dictionaries.

Dropbox has published a blog post explaining their OCR technology achieved using Computer Vision and Deep Learning. They have created this system specifically for mobile handheld devices. The first task was to label a picture data. This will be used for training the models later on. The labeling was done on user collected data by human intervention. The labeling was performed by specific teams at DropTurk and MTurk. This labeling is considered as a ground truth for individual images. The next job is to detect words from an image. This problem is divided into 2 parts. That is dividing the images into images containing one line per image and using these images to get actual text using Deep Neural Networks. Images are fed to Convolutional Neural Network. The output of CNN is then fed to Bidirectional Long short Term Memory Which is used for speech recognition and consequently, making words from range of characters.

The main problem of using deep learning is that the neural nets typically require a huge amount of dataset for achieving good accuracy. Getting original data from user is difficult. Dropbox used another Deep Learning model called Generative Adversarial Models which is well suited for generating realistic data. All this process gave more accuracy for commercial use on one single word. To get the system working for entire document, the paragraphs of words are broken down into separate lines for input to the neural nets. This is achieved by Maximally Stable Extremal Regions using OpenCV's implementation. The models were first developed in Torch but were imported to Tensorflow. [2]

Frameworks

Communication between applications allow the application to provided extended functionality. The applications can communicate using messaging, thus making them loosely coupled. This loose coupling, eliminates the requirement of running the applications at the same time. Messaging makes the messaging system responsible for transferring data from one application to another, so the applications can focus on what data they need to share as opposed to how to share it. It's a set of connections that enables applications to communicate by transmitting information in predetermined, predictable ways [3].

The amount of data being generated is increasing tremendously. Business value can be derived from the information by analyzing. Analysis of this ever-growing data becomes a challenge with traditional analytical tools. In order to be able to analyze the collected information, scalable, high performing and flexible tools are required. However, organizations are facing a growing big data ecosystem where new tools emerge and "die" very quickly. Therefore, it can be very difficult to keep pace and choose the right tools. [4]

Kafka

Linkedin's log data processing application required processing large amount of information to track user engagement and system utilization. The information could be processed offline and may also need to incorporate the real time information added to system. Traditional messaging systems added overhead by providing delivery guarantees which is not required for log processing, provided no support for distributed systems and did not focus on throughput. In order to address these challenges, LinkedIn developed a messaging system for processing logs called Kafka. It combined the log aggregator application along with messaging systems. Kafka is distributed and scalable and offers high throughput. It also provides an API similar to a messaging system and allows applications to consume log events in real time. Kakfa is a distributed platform which processes streams of data as they occur. Each stream is characterized by a topic. A producer can publish messages to a topic, which are stored on a set of servers called brokers. Consumers can subscribe to one or more topics from the brokers and consume the subscribed messages by pulling data from the brokers. Multiple brokers form a cluster and each broker stores one or more partitions of a topic. Multiple producers and consumers can publish and retrieve messages at the same time. [5]

Apache Spark

Spark was introduced as a framework that works efficiently for data intensive applications, similar to Hadoop. One of the primary disadvantage of Hadoop is that it uses an acyclic data model that is suitable for only a few types of applications. Spark intends to target the applications that reuse a working data set across multiple parallel operations. Spark is usually used for machine learning algorithms where iteratively, the algorithm will read the same data set over and over. Hadoop performs poorly in this case because every time, the data has to be loaded from the disk and processed. Spark uses an abstraction called Resilient Distributed Dataset(RDD), which is similar to a shared memory (read-only) but actually is not. RDDs are scalable and fault-tolerant like MapReduce. RDD can be built in 4 ways: 1) From a file, 2) By parallelizing a Scala collection, 3) By transformations of existing arrays and 4) By changing the persistence of RDDs on disk(cache and save operations). The main parallel operations of Spark are Reduce(transforms an array into another), Collect (sends the elements of the RDD to a driver program), foreach(passes each element into a user-provided function). A limited set of shared variables are allowed in Spark. They are Broadcast variables – instead of sending these variables in every function, each worker will receive one copy of the variable and will work on it - and Accumulators – variables only the driver program can read, typically, a counter. Spark typically achieves performance of querying a DB with 50GB data in under a second. The initial version of Spark was built only on the Scala platform.[6]

Spark SQL

To support relational database support into Apache Spark, a modified version called Shark was initially introduced. It was later observed that Shark suffered from many disadvantages like restricting the users to choose either relational API or procedural API but not both. Another scenario in which Shark performed badly was in machine learning algorithms where the users might want to perform relational operations of already computed results instead of the actual data on the database. Spark SQL overcomes these cons in two ways. Firstly, it provides a Dataframe API which intermixes relational and procedural APIs allowing higher flexibility to the users. Additionally, it provides an optimizer called Catalyst that makes it easy to add dynamic data into the database. A dataframe can be visualized as a table in a relational database. These tables can be built on both the RDDs and on the external data. The dataframes are not physically executed until an output operation of a query is executed. The Spark SQL datamodel supports both basic SQL data types like int, float, string etc., complex data types like arrays, maps and unions, and user-defined data types as well. The relational operations on dataframes are performed using a Domain Specific Language (DSL). The catalyst optimizer was built to support extension of spark APIs for the semi-structured data types and to help external users to generate data-source specific rules that could push filtering or aggregation into external storage as well. The basic data type in catalyst is a tree of node objects. Each node has a node type and zero or more children. New node types are implemented as subclasses of the treenode base class. Rules are functional transformations that replace one tree with another. Pattern Matching is used to replace the entities in a given rule with the actual nodes of a tree for the transformation.[7]

Location based Recommendation Services

Our application needs a large amount of database of stores across USA. This makes the information complex and hard to filter. We are considering an addition of location based filtering. This will allow the user to get the store data that is nearer to them faster, and efficiently. This will also allow us to implement pub-sub system for users favorites and shopping lists.

Here we discuss some research done in this area. The first paper implements a recommendation system based on location. Recommendation system can consider user's location for recommending the nearest service (ATM, restaurants, pubs, tourist sights and social events). The MOPSI project implements various location-based services and applications such as mobile search engines, data collection, user tracking and route recording. It has applications integrated both on web and in mobile phones.

Aim in MOPSI is to recommend interesting places in user's surrounding. The system gives personalized recommendations by combining various paradigms of recommendation systems. It uses location as a pre-selection criteria. The data that is present far away from the user is irrelevant. After that the sorted list is created from the pre-selection. Services are scored using contextual information about search history, location, and explicit rating. Various types of scoring functions are used for calculating proximity scores based on user location.[8]

References

[1] A Survey on Optical Character Recognition System by Noman Islam, Zeeshan Islam, Nazia Noor
https://arxiv.org/ftp/arxiv/papers/1710/1710.05703.pdf

[2]https://blogs.dropbox.com/tech/2017/04/creating-a-modern-ocr-pipeline-using-computer-vision-and-deep-learning/

[3]Enterprise Integration Patterns: Designing, Building, and Deploying , Gregor Hohpe, Bobby Woolf

[4]A Framework for Real-time Streaming Analytics using Machine Learning Approach, Ms.D.Jayanthi, Dr.G.Sumathi

[5]Kafka: a Distributed Messaging System for Log Processing, Jay Kreps, Neha Narkhede, Jun Rao

[6]Spark: Cluster Computing with Working Sets. Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica. HotCloud 2010. June 2010.
http://people.csail.mit.edu/matei/papers/2010/hotcloud_spark.pdf

[7]Spark SQL: Relational Data Processing in Spark. Michael Armbrust, Reynold S. Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K. Bradley, Xiangrui Meng, Tomer Kaftan, Michael J. Franklin, Ali Ghodsi, Matei Zaharia. SIGMOD 2015. June 2015.
http://people.csail.mit.edu/matei/papers/2015/sigmod_spark_sql.pdf

[8]Context Aware Recommendation of Location-based Data
https://www.researchgate.net/profile/Andrei_Tabarcea2/publication/241193539_Context_aware_recommendation_of_location-based_data/links/0c9605322b341b1b73000000/Context-aware-recommendation-of-location-based-data.pdf?origin=publication_detail