International Journal of Advance Engineering and Research Development (IJAERD)
Special Issue, Volume 1,Issue 4, April 2014, e-ISSN: 2348 - 4470 , print-ISSN:2348-6406

Product Recognition API

Kishan Thobhani[1], Deep Patel[2], Asst. Prof. Vatsal Shah[3]

[1]*Information Technology Department, BVM Engineering College, thobhanikishan@gmail.com*
[2]*Information Technology Department, BVM Engineering College, deeep580@gmail.com*
[3]*Information Technology Department, BVM Engineering College, vatsal.shah@bvmengineering.ac.in*

Abstract— An API solution for product recognition. A set of libraries will be exposed online for other clients to query for their product recognition needs. After taking picture of a product referenced in sources like newspaper, posters etc., the user can retrieve specific information embedded in the product by querying the API. Our server will be running machine-learning algorithms pertaining computer vision by running several tasks related to semi-accurate technologies like ACR & OCR to text recognition. This on later stage will be used to enhance experience of algorithm as well as performance. A pre-set of dataset will be maintained in abstract neural network that will be broken down in set of data itself by algorithm to map with query fed and return output in abstract way as well as storing for future needs. This is similar to how human brain process information.

Keywords- Product, query, caching server, algorithmic, mapping

INTRODUCTION

A scalable intelligent APIs to recognize products on fly. Consider a client wants a digital signage solution to embed information inside pre-existing posters, signboards, billboards etc. without modifying it or re-printing it with information embedding technologies like QR code or Bar code. For this a Product recognition API will be exposed that client could call by sending multiple parameters of information like image, location using Internet medium and our server using intelligent algorithm retrieves information for it and outputs it to client. At the same time storing it for the future queries.

There will also be a caching layer responding to the Meta data like location to respond to the query quickly provided information is cached initially. If not so the caching server will pass the request to the algorithmic server to do the further processing. Note caching server acts as a layer for doing all the per-processing tasks like creating an authentication. Histogram/character recognition. Thus creating a first layer of abstraction to be fed to the algorithmic server.

There are several apps that do similar tasks like: Kooba & Google Goggles [1]. But we are creating a third party solution to crater need for advance content or brand recognition. Note on the backend our existing dataset has been curated/derived taking in account of the apps mentioned above. So results to queries will be close to them or better than them considering the efficiency and accuracy of the learning algorithm.

Also the API takes in account of response time that has been greatly optimized using several compressing techniques. Also by creating an abstract query and patterns our algorithm instead of traversing the whole network tries to map pattern to pre-connected nodes or pre-established pattern as on caching layer.

I. SYSTEM OVERVIEW

API will be exposed to clients from a scalable cloud. Its will be HTTP API (Reprentational state transfer) [2].

Authentication will be managed by a OAuth [3] protocol which is an open standard for authorization. OAuth provides a method for clients to access server resources on behalf of a resource owner such as a different client or an end-user.

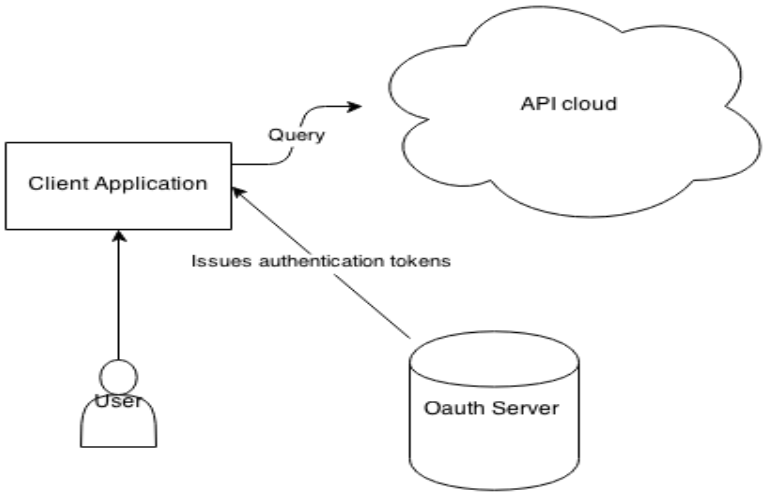

Figure 1 Authentication Mechanism

Clients connecting to API will have following responsibilities:

- Registering Application with API
- Creating a dataset for mandatory parameters like imaging, location etc.
- Packaging in-form of a REST Request
- Querying

Whole neural network [4] of global data will be index by locations and thus creating a mind map.

Figure 2 Location subset neural network level 1

2.1 Caching Server

Caching server has multiple roles. Initially it checks the location feed and tries to narrow down feed to specific mind map. Later it tries to establish patterns based on tree data-structure and creates abstract information using various compression techniques. It than compares this information with the cached results. If found it will return it on the go. Or else it will pass this information to the latter algorithmic instance.

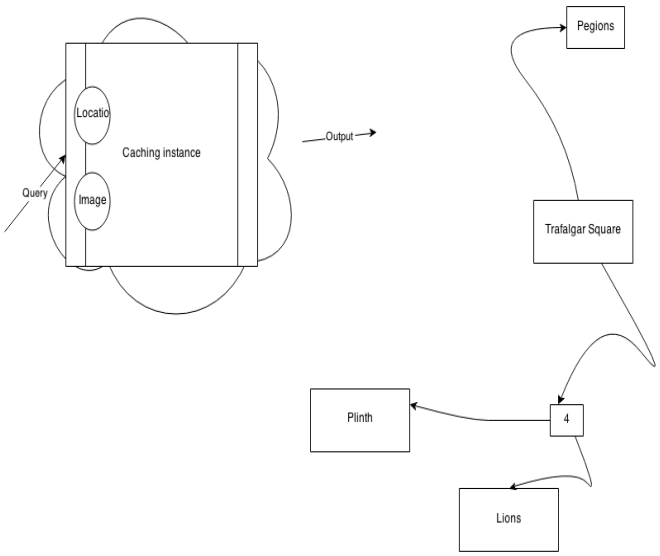

Figure 3 Creating a mind-map to be mathematically mapped to neural network

2.2 Algorithmic Instance

This will be sharing a global neural network. Each output from the caching server will be broken down into number of instances based on various patterns thus creating a mind maps. This mind maps will be mapped to neural networks across various instances and running multiple threads to speed up the execution. Once the pattern is found mapped it will derive information stored in the database by indexing the nodes. This information will be merged with the multiple outputs that are derived from across various threads and processed and patterned again thus creating connected nodes at particular instances.

For e.g. in our Trafalgar square mind map.

- Pigeons will be mapped with types of pigeons.
- Number 4 will be mapped with the cluster of 4 things.
- Plinths will be mapped to the architectural cluster
- Lions will be mapped to animals' kingdom.
- Multiple mapping of lions and plinths will result in a pattern in architectural cluster.
- Thus latter the numbers, pigeons, plinths and lions will form a pattern in itself and will be connected to each of this individual cluster.

This way algorithm will enhance its experience and maps will result in performance enhancement.

II. SYSTEM IMPLEMENTATION

A product recognition No-SQL database of more than one billion nodes forming a basic neural network is made initially. This database is collected from the services mentioned earlier like Goggles, Image search, Google Maps etc.

Deploy over cloud with a proxy server running caching server as well as load balancing across other algorithmic instances. This is done on a Linux server with a x-large GPU EC2 instance on AWS.

3.1 Time

This is taken in account to transmission time & processing time. Transmission time solely relies on the network connectivity. Processing time or performance will be retrieved based on the desired accuracy of the algorithm that would be based on the experience. But in ideal manner it shouldn't be more than 8s.

3.2 Compression

This is a post abstraction feature on the caching layer. Compression has to be lossless and can be based on features that are retrieved from the imaging. Also the whole compression has to be searchable and index able.

REFERENCES

[1] Sam S. Tsai, David Chen, Vijay Chandrasekhar Gabriel Takacs, Ngai-Man Cheung, Bernd Girod, Ramakrishna Vedantham Radek Grzeszczuk Nokia Research Center in paper "Mobile Product Recognition".
[2] Subbu Allamaraju in "RESTful Web Services Cookbook"
[3] Martin Spasovski in "OAuth 2.0 Identity and Access Management Patterns"
[4] M. Egmont-Petersena, D. de Ridderb, H. Handelsc in paper "Image processing with neural networks"
[5] P.S.Suhasani , Dr. K.Sri Rama Krishna, Dr. I. V. Murali Krishna in paper "Cbir Using Color Histogram Processing".

