JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Using Recurrent Neural Network for Taxi Demand Prediction

Asst.Prof S. Revathi
dept. Information Technology
Aalim Muhammed Salegh College of Engineering
Chennai, India
revathi.s@aalimec.ac.in

Sanjay Bitra
dept. Information Technology
Aalim Muhammed Salegh College of Engineering
Chennai, India
bitrasanjay43@gmail.com

Saba Zehra Syed
dept. Information Technology
Aalim Muhammed Salegh College of Engineering
Chennai, India
syedsaba1974@gmail.com

***Abstract*—The taxi administration sector has been booming in recent years, and it is expected to continue to grow shortly. Cab drivers must decide where to stop for passengers so that they can pick someone within a limited waiting time. Travelers also like a quick cab service when they need it.The taxi administration's control center determines the most congested area of focus.Cabs were spread over scattered demography in the current framework,over a larger area without a time-based inhabited zone such as an airport,commercial zones, school zones, train stops, and so forth.Effective taxi assignments can assist drivers and passengers in reducing the time it takes to locate one another. Future interest can be predicted in the proposed framework using a Recurrent Neural Network-based model that can be developed with historical data. By resolving taxi accessibility, it will be able to service more clients in a brief period. The data collection includes the taxi's GPS location as well as various taxi characteristics such as slanted edge, pickup spot, and so on. This model is used to predict interest in various areas of the city over a given time frame.**

Keywords—taxi-passenger demand,Global Positioning System (GPS) data, Recurrent Neural Networks.

I. Introduction (*Heading 1*)

The ultimate goal of the script is to predict the high demand for pickup sites for taxi services based on their historical experience. During this paper, we will be looking at two key algorithms: pre-processing and RECURRENT Neural Networks.

We'll simply clean the data from the info set during pre-processing. It is conventional to separate the desired knowledge from the undesirable knowledge. With historical knowledge, the RECURRENT Neural Network-based model is trained. Then, during the next 6 hours, we'll construct a graph that will help us predict accurately.Given the enormous number of techniques available, we have chosen to use two primary algorithms: pre-processing and RECURRENT neural network (RNN). Preliminay process of informations to prepare it for further analysis by the primary processor. This term refers to any initial process stage in which multiple steps are required to organize knowledge for the user. A RECENT Neural Network (RNN) is a Deep Learning rule that processes incoming images and is capable of distinguishing one image from others. When compared to other classification methods, the amount of pre-processing required is significantly less.

The impact of transportation planning and solutions on city life is immense. To reduce travel time, transportation service providers such as Didi and Uber must first comprehend and elaborate on a city's mobility, which is emerging more and more likely to be achieved as the Internet of Vehicle grows in prominence.However, we were still inconvenienced by the long wait for a taxicab. As a result, figuring out how to maximize the use of these taxis while reducing passenger wait times is a pressing concern. The traditional issue of passenger modeling for taxi and prediction always uses GPS trajectory data, with time series demonstrating spatial dependencies, which are found in many fields such as economics, transportation, and vision. In taxi-recommender systems, there are a variety of statistical learning models for passenger prediction. However, due to complicated feature engineering, these approaches always have poor performance and are not suited for real-time systems.Furthermore, generalizing these models to new applications is difficult. The neural network has demonstrated its superiority in autonomous learning with the rise of machine learning. Deep learning has spawned a slew of models that may be applied to urban science and the Internet of Vehicles to capture important data and perform difficult job sequences. These models can extract relevant information from data, but the spatial organization, which is critical in traffic applications, is rarely taken into account. In [6,7,8] authors aim to maximize drivers' profits by providing routing recommendations .These works give valuable outputs ,but they only consider the current passenger requests and available taxis

To overcome these drawbacks, we developed L-RNN, a unique real-time taxi passenger prediction algorithm based on the Cortex net.L- RNN uses a high-level architecture that includes an online learning framework and a novel neural network based on Convolutional Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and the Embedding layer.

The purpose of RNN is to understand and elaborate passenger mobility directly from input comparable to the image. LRNN is a novel traffic prediction system that draws inspiration from image processing technologies.

a) Propose native L-RNN natural networks built on LSTM, RNN, and embedding layer to forecast possible passengers for taxi drivers without requiring considerable feature engineering or other external data.Meanwhile, for taxi-passenger prediction, the system can collect information on both the temporal and spatial dimensions, and it can be simply adapted to other traffic prediction issues.

b) For real-time prediction, an online learning approach is proposed. The model can be trained to maintain the state current based on the most recent data.

As a result, it's appropriate for the current popular streaming data prediction method. It eliminates the need to save historical data because all knowledge is stored in the model. Intelligent transportation systems for efficient taxi dispatching [1], time-saving route finding [2], [3], fuel-saving routing [4], and taxi sharing [5] are already successfully exploring these kind of data and/or interfaces

The basic topic of passenger modelling for taxi and prediction has generated a large body of work in statistics and machine learning. Taxicabs have recently been equipped with GPS sensors in numerous major cities, including New York, Beijing,Chennai, Bengaluru, and Chengdu, resulting in a large number of GPS trajectories with occupancy information being created every day. Several works have effectively examined this type of data with a variety of applications, including smart driving and modelling the spatiotemporal structure of taxis.

II. Motivation

Taxi drivers must decide where to wait for passengers so that they can be picked up as quickly as possible. Passengers also prefer to locate a taxi as soon as they are ready for pickup. The taxi service's control center determines the congested location to be concentrated. The taxis were sometimes dispersed across greater demography, missing the time-based busy area such as an airport, a business district, a school district, or a train station, etc.Managing fleet of taxis to a crowded area.The increased need for effective utilization of resources to reduce waiting time for passengers,serve more customers in a short time by organizing the availability of taxis is the motivation behind this research.

III. Approach

(Fig 1) Effective taxi dispatching can reduce the time it takes for drivers and passengers to find each other because drivers do not have adequate information about where passengers and other cabs are and intend to travel, a taxi center can arrange the taxi fleet and distribute it efficiently based on demand throughout the city. To construct such a taxi hub, an intelligent system that can

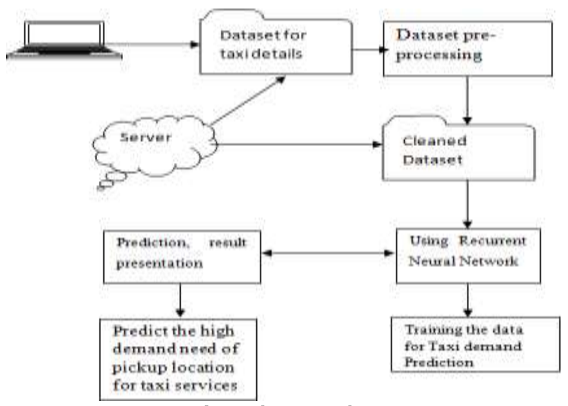

Dataset for taxi details
Dataset pre-processing
Server
Cleaned Dataset
Prediction, result presentation
Using Recurrent Neural Network
Predict the high demand need of pickup location for taxi services
Training the data for Taxi demand Prediction

Fig 1: Taxi dispatch approach.

forecast /predict future demand across the city is required. To predict future demand, our system uses GPS position and other taxi features such as drop point, pickup point, and so on. Using historical data, a RECURRENT Neural network-based model is trained.This model is used to predict the demand in different areas of the city.

IV. Machine learning

Machine learning (ML) is the study of algorithms and statistical models that computer systems use to complete a task without utilizing explicit instructions, instead of relying on patterns and inference. It is considered a subset of AI stands for artificial intelligence. Algorithms for machine learning create a mathematical model based on a sample of data, referred to as "training data," to create predictions or forecasts decision-making without being explicitly programmed to do so. Machine algorithms for learning are employed in a wide range of applications, including email filtering.In computer vision, where developing a traditional system is a difficult or impossible algorithm for completing the task.Computational statistics and machine learning are closely related focuses on using computers to make predictions. The investigation of methods, theory, and application domains are all available through mathematical optimization.

V. Machine Learning Tasks

Machine learning tasks are classified into several broad categories. The procedure creates a mathematical model from a set of data that includes both the inputs and the expected outputs in supervised learning. For Example, If the aim was to determine whether an image contained a specific object, the training data for a supervised learning algorithm would include images with and without the object (the input), and each image would have a label (the output) indicating whether it did or did not contain the object. In other circumstances, the input may only be available in part or be limited to specific feedback. Semi algorithms create mathematical models from partial training data, in which some of the sample input has no labels.

Supervised learning methods include classification and regression algorithms. When the outputs are limited to a small set of values, classification methods are applied. An incoming email would be the input to a classification algorithm that filters emails, and the output would be the name of the folder to file the email in. The output of a spam email detection algorithm would be a prediction of "spam" or "not spam," represented by the Boolean values true and false. Regression algorithms get their name from the fact that their outputs are continuous, meaning they can be any value within a range. Temperature, length, and object price are all examples of continuous values.

The procedure creates a mathematical model from a set of data that comprises only inputs and no desired output labels in unsupervised learning. Unsupervised learning techniques are used to discover structure in data, such as data point grouping or clustering. Unsupervised learning, like feature learning, can find patterns in data and organize inputs into categories. The practice of lowering the number of "features," or inputs, in a piece of data is known as dimensionality reduction.

Active learning algorithms access desired outputs (training labels) for a limited set of inputs based on a budget and optimize the inputs for which training labels will be acquired. These can be supplied to a human user for labeling when utilized interactively.

Reinforcement learning algorithms are employed in autonomous vehicles or when learning to play a game against a human opponent and receive feedback in the form of positive or negative reinforcement in a dynamic environment.

Topic modeling is another specialized approach in machine learning, in which computer software is given a set of natural language texts and is asked to locate more documents that cover comparable themes. In density estimation challenges, machine learning algorithms can be utilized to locate the unobservable probability density function. Based on prior experience, meta-learning algorithms learn their own inductive bias.

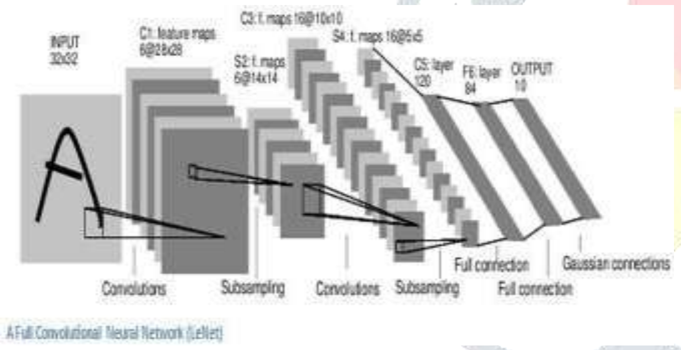


Fig 2: A full Convolutional Neural Network

VI. Types of Learning algorithms

Machine learning algorithms differ in terms of their approach, the type of data that input and output, and the task or problem they are intended to address.

A. *Supervised learning*

Supervised learning algorithms construct a mathematical model of a set of data that includes both the inputs and the outputs that are desired. The information is referred to as training data, and it consists of a collection of training instances. Each training example has one or more inputs and a supervisory signal as the desired output. Each training sample is represented by an array or vector, sometimes referred to as a feature vector, and the training data is represented by a matrix in the mathematical model.

Supervised learning techniques develop a function that may be used to predict the output associated with fresh inputs by iteratively optimizing an objective function. The algorithm will be able to accurately estimate the output for inputs that were not part of the training data if it uses an optimum function.Classification and regression are examples of supervised learning techniques.When the outputs are confined to a limited multitude of alternatives, classification methods are applied.

B. *Unsupervised Learning*

Unsupervised learning methods take a collection of data with only inputs and identify structure in it, such as data point grouping or clustering. As a result, the algorithms learn from unlabeled, unclassified, and uncategorized test data. Unsupervised learning algorithms discover commonalities in the data and react depending on the existence or lack of such commonalities in each new piece of data, rather than responding to feedback. Unsupervised learning is commonly used in the field of statistics for density estimation, but it also applies to other domains that need summarizing and explaining data aspects.

C. *Semi-Supervised Learning*

Unsupervised learning (without any labeled training data) and supervised learning (with labeled training data) are the two types of learning (with completely labeled training data).Many machine-learning researchers have discovered that combining unlabeled data with a tiny quantity of labeled data can result in a significant boost in learning accuracy.

D. *Recurrent Neural Network*

(Fig 2) Recurrent neural networks (RNN) may sound like a unique combination of biology and math with a dash of computer science thrown in, but they've been some of the most impactful developments in the field of computer vision. Alex Krizhevsky used neural nets to win the ImageNet competition (essentially, the yearly Olympics of
computer vision) in 2012, decreasing the classification error record from 26% to 15%, an astonishing improvement at the time. Since then, a slew of businesses has incorporated deep learning into their offerings.

Facebook's automated tagging algorithms are based on neural networks, while Google's photo search, Amazon's product suggestions, Pinterest's home feed personalization, and Instagram's search architecture are all based on neural networks.However, image processing is the original and possibly most prominent application of neural networks.

VII. The problem space

Image classification is the process of taking an input image and determining which class (cat, dog, etc.) or probability of classes best describes it. Recognition is one of the earliest talents that humans learn from the moment they are born, and it comes readily and effortlessly to adults.
We can rapidly and effortlessly identify the surroundings we're in, as well as the objects that surround us, without even thinking about it. Most of the time, whether we see an image or simply gaze at the environment around us, we can characterize the scene and assign labels to each object without even realizing it.
We don't share these abilities to recognize patterns rapidly, generalize from existing information, and adapt to new visual settings with our fellow machines refer Fig 3.

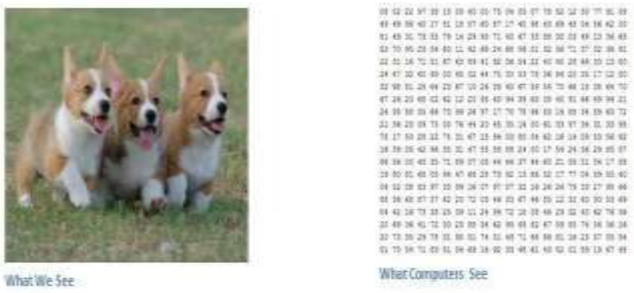


Fig 3: The problem space (left :what we see ,right :what computers see)

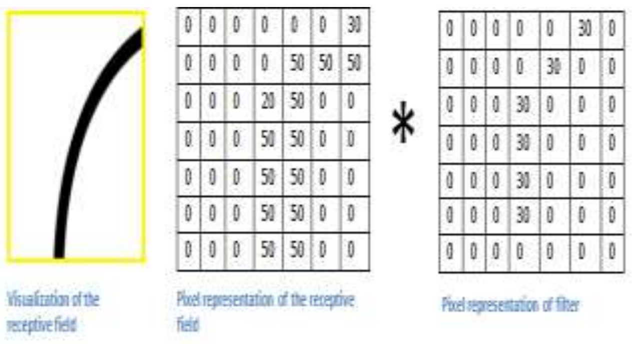
A. *Inputs and Outputs*

A computer will see an array of pixel (Fig2) values when it views an image (takes an image as input). It will see a 32 × 32 x 3 array of numbers, depending on the image's resolution and size (The 3 refers to RGB values). Let's pretend we have a color image in JPG format with a resolution of 480 by 480 pixels. 480 × 480 x 3 will be the representative array. Each of these numbers is assigned a value between 0 and 255 that describes the pixel intensity at that particular location. These numbers are the only inputs available to the computer, and while they are worthless to us while performing image classification, they are the only inputs available to the computer. The concept is that you give the computer this array of numbers, and it returns values that describe the likelihood that the image belongs to a specific class (.80 for cat,.15 for dog).

VIII. THE MATH BEHIND

A RECURRENT Layer is always the first layer in an RNN. The first thing to understand is what this conv (I'll be using that term a lot) layer's input is.(Fig 3) The input is a 32 × 32 x 3 array of pixel values, as noted previously. Imagine a flashlight flashing over the top left corner of the image to show what a conv layer is. Let's imagine the light emitted by this flashlight spans a 5 × 5 space. Now imagine that this flashlight is sliding across the entire input image. This flashlight is known as a filter (or sometimes referred to as a neuron or a kernel) in machine learning, and the region it shines across is referred to as the receptive field. This filter is now also a number array (the numbers are known as weights or parameters). It's vital to notice that the depth of this filter must match the depth of the input (this ensures that the math works), thus the filter's dimensions are 5 x 5 x 3. Let's look at the first place of the filter as an example. It's in the upper left corner. The filter multiplies the values in the filter with the original pixel as it slides around the input image, or convolved, image's parameters (aka computing element-wise multiplications). All of these multiplications are added together (totaling 75 multiplications in mathematics). So you've got a single number now. Remember that this value only applies when the filter is in the upper left corner of the image. This step is now repeated for each position on the input volume. (The next stage would be to move the filter 1 unit to the right, then 1 unit to the right again, and so on.) A number is generated for each unique location on the input volume. After you've slid the filter over all of the places, you'll be left with a 28 × 28 x 1 array of integers, which we refer to as an activation map or feature map. The rationale for the 28 x 28 array is that a 5 x 5 filter can only fit 784 distinct spots fit on a 32 x 32 input image. These 784 numbers are mapped to a 28 x 28 array.

Fig 3: Visualization of 5X5 convolving around an input volume and producing an activation map.


Visualization of the receptive field
Pixel representation of the receptive field
Pixel representation of filter

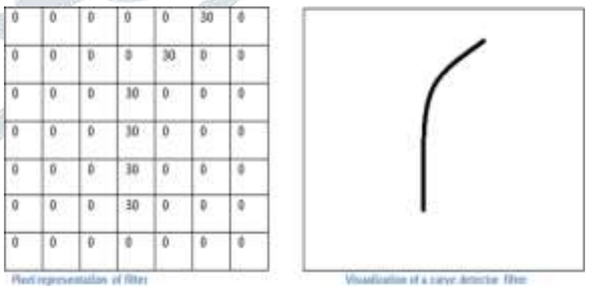
Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

Let's assume instead of one 5 x 5 x 3 filter, we now use two 5 x 5 x 3 filters. The volume of our production would therefore be 28 × 28 x 2. We can better retain the spatial dimensions by applying more filters. This is what happens in a RECURRENT layer mathematically.

A. *High-level prognosis*

But first, let's look at what this convolution (Fig 4) is doing at a high level. These filters can all be considered feature identifiers. I'm referring to things like straight lines, plain colors, and curves when I mention features. Consider the most basic traits that all photos share. Let's imagine our first filter is a curve detector with dimensions of 7 x 7 x 3. (For the sake of simplicity, disregard the fact that the filter is three units deep in this part and only consider the top depth slice of the filter and the image.) The filter will have a pixel structure as a curve detector, with greater numerical values along the area that is a curve shape (remember, these are only numbers!).Let's return to the mathematical representation of this now.

Fig 4: High level perspective(Left: pixel representation of filter, Right:


Pixel representation of filter
Visualization of a curve detector filter

visualization of curve detector filter.)

When we place this filter in the input volume's top left corner, it computes multiplications between the filter and the pixel values in that area. Let's use an image that we wish to categorize as an example, and place our filter in the top left corner.Remember, what we have to do is multiply the values in the filter with the original pixel values of the image.

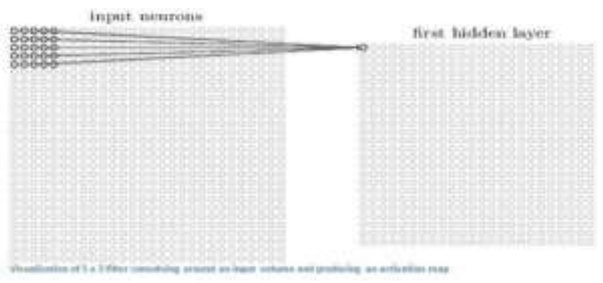

input neurons
first hidden layer
Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

Fig 5: Pixel representation(Left:Visualization of the respective field, Centre:pixel representation of the respective field, Right:pixel representation of filter)

If there is a shape in the input image that resembles the curve that this filter represents, then adding all of the multiplications together will yield a significant value! Let's have a look at what occurs when we move our filter around.The price is a lot lower! Because nothing in the image(Fig5) area responded to the curve detector filter, this was the case. Remember that the activation map is the result of this conv layer.

So, in the simple example of a single filter convolution (and if that filter is a curve detector), the activation map will reveal the parts of the image where curves are most likely to appear. The top left value of our 26 x 26 x 1 activation map will be 6600 in this case (26 because of the 7x7 filter instead of 5x5).

This high value indicates that the filter (Fig 6) was activated as a result of some form of the curve in the input volume. Because there was nothing in the input volume that caused the filter to activate (or, to put it another way, there wasn't a curve in that part of the original image), the top right value in our activation map will be 0. Keep in mind that this is only for one filter. This is simply a filter for detecting lines that curve outward and to the right. For lines that curve to the left or straight edges, we can use different filters. The depth of the activation map increases as the number of filters increases, giving us more information about the input volume.

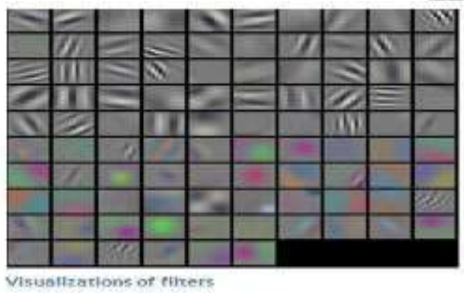


Fig 6: filter

IX. Dataset Preparation

We'll get our datasets from Kaggle, which is the largest platform for data scientists and machine learning practitioners, and so provides the finest practical experience to the difficult field of data science for aspirants. Python is our preferred language because it offers a large library of machine learning libraries. There may be empty values, negative values, or errors in the dataset. During pre-processing, the dataset is cleansed. The pre-processing methods entail the removal of incomplete records. Once we have a clean dataset, we must prepare it for use by the machine learning algorithm.

A. Dataset pre-processing

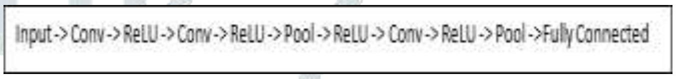
Cleaning, instance selection, and normalization are all examples of data pre-processing.The final training set is the result of data pre-processing, which includes transformation, feature extraction and selection, and so on. The way data is pre-processed can have an impact on how it is used.It is possible to interpret the results of the final data processing.

B. Cleaned Dataset

The practice of correcting or deleting incorrect, corrupted, improperly formatted, duplicate, or incomplete data from a dataset is known as data cleaning. Even if the data is right, outcomes and algorithms are untrustworthy if the data is erroneous.

C. Training dataset

The data you use to train an algorithm or machine learning model to anticipate the outcome you want it to predict is known as training data. Test data is used to evaluate the algorithm you're using to train the machine's performance, such as accuracy or efficiency. Following the collection of training data, you proceed to the following stage of machine learning: Data preparation is the process of loading data into an appropriate location and then preparing it for machine learning training.



Input->Conv->ReLU->Conv->ReLU->Pool->ReLU->Conv->ReLU->Pool->Fully Connected

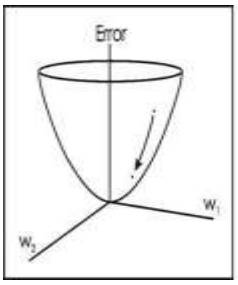
Fig7: Training of model

X. Training the Model

(Fig 7) Back propagation is a training technique that allows the computer to change its filter values (or weights).Before we get into back propagation, let's take a step back and discuss what a neural network requires to function properly. Our thoughts were all blank when we were all born. We had no idea what a cat, dog, or bird was. The weights or filter values are randomized similarly before the RNN starts. The filters aren't aware of the need of looking for edges and curves. The filters in the higher layers aren't as effective for looking at paws and beaks.Our parents and teachers, on the other hand, as we grew older, showed us various photos and images and labeled them accordingly.

RNNs go through a training procedure in which they are given an image and a label. Let's pretend we have a training set with thousands of images of dogs, cats, and birds, each with a label indicating which animal it belongs to. Prop back to back.The forward pass, the loss function, the backward pass, and the weight update are the four distinct portions of back propagation. However, this does not indicate a predilection for any particular number. With its current weights, the network is unable to look for those low-level properties, and so is unable to draw any valid conclusions about the categorization. This is related to back propagation's loss function. Keep in mind that we're still working with training data. There is an image and a label for this data. Let's say the first training image that was entered was a [3. 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] A loss function can be described in a variety of ways, but the MSE (mean squared error) is a typical one, which is 12 times (actual - predicted) squared.

Assume that the variable L has the same value. For the first handful of training images, the loss will be exceedingly high, as you could expect. Let's think about this intuitively now. We want to get to the point where the predicted label (ConvNet output) matches the training label (indicating that our network predicted correctly). We want to keep our losses to a minimum to get there. We want to find out which inputs(weights in our example) contributed the most directly to the network's loss (or error) by visualizing this as a calculus optimization problem.



One way of visualizing this idea of minimizing the loss is to consider a 3-D graph where the weights of the neural net (there are obviously more than 2 weights, but let's go for simplicity) are the independent variables and the dependent variable is the loss. The task of minimizing the loss involves trying to adjust the weights so that the loss decreases. In visual terms, we want to get to the lowest point in our bowl shaped object. To do this, we have to take a derivative of the loss (visual terms: calculate the slope in every direction) with respect to the weights.

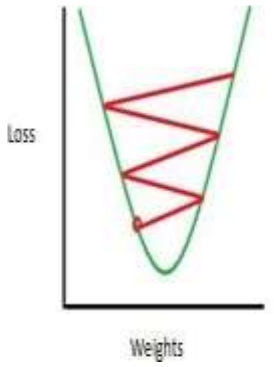
(Fig 8: Derivative of loss diagram)

This is the mathematical equivalent of a dL/DW, where W denotes the layer weights.(Fig8) Now we want to go back through the network, discovering which weights contributed the most to the loss and figuring out how to alter them such that the loss is reduced. We then proceed to the final step, which is the weight update, after computing this derivative move in the opposite direction of the gradient.

$$w = w_i - \eta \frac{dL}{dW}$$

w = Weight
w_i = Initial Weight
η = Learning Rate

The learning rate is a parameter that the programmer selects. A high learning rate means that weight changes are made in larger stages, and the model may converge on an ideal set of weights in less time. A high learning rate, on the other hand, may result in jumps that are too large and imprecise to reach the optimal position(Fig9).



Consequence of a high learning rate where the jumps are too large and we are not able to minimize the loss.

Fig 9:Consequence of high rate learning

One training iteration includes forward pass, loss function, backward pass, and parameter update. For each set of training images, the program will repeat this process for a predetermined number of iterations (commonly called a batch). Once you've finished updating the parameters on the last training example, the network should be well-trained enough that the layer weights are calibrated correctly.

XI. Prediction and Result

A graph is plotted for the future prediction for the next time slot and the area to be crowded. This machine learning model predicts the future demand area in a city based on and the drivers were taken to wait in the area where the system identified as demand area.

XII. Conclusion

In this script, we look into the problem of predicting taxi passengers in real-time. One of the greatest approaches to predict travelers is to use processing and a recurrent neural network (RNN). We can forecast the number of people ahead of time, saving both fuel and time for the customer. It is one of the most effective approaches for attracting clients and generating profit for the organization.Effective taxi dispatching will reduce the time it takes for each driver and passenger to locate one another. Drivers do not have adequate information on where passengers and other cabs are and will go. As a result, a taxi center will arrange the taxi fleet and distribute them efficiently following the demand from the entire municipality. To run a taxi center like this, you'll need an intelligence system that can forecast long-term demand over the entire town.

To forecast long-term demand, our system analyses GPS position and other taxi attributes such as drop purpose, pickup purpose, and so on. With historical data, a RECURRENT neural network-based model is trained. This model is used to forecast demand in a variety of places around town.

XIII. Future Enhancement

Due to its testbed, this approach has no peer in the literature; the models have been evaluated in a streaming context, whereas the state of the art primarily covers offline experimental settings. This model will be utilized as a part of a recommendation system that will provide taxi drivers with smart real-time advice about which taxi stand they should go to after a drop-off.

XIV. Abbreviations

RNN - Recurrent Neural Network

LSTM - Long Short-Term Memory

CNN - Convolution Neural Network

RAM- Random Access Memory

UML - Unified Modelling Languages

$$E_{total} = \sum \frac{1}{2}(target - output)^2$$

REFERENCES

[1] A. Glaschenko, A. Ivaschenko, G. Rzevski, and P. Skobelev, “Multi-agent real time scheduling system for taxi companies,” in Proc. 8th Int. Conf. AAMAS, Budapest, Hungary, 2009, pp. 29–36.

[2] J. Lee, G.-L. Park, H. Kim, Y.-K. Yang, P. Kim, and S.-W. Kim, A TelematicsServiceSystemBasedontheLinuxCluster. Berlin,Germany: Springer-Verlag, 2007, pp. 660–667.

[3] J. Yuan, Y. Zheng, X. Xie, and G. Sun, “T-drive: Enhancing driving directions with taxi drivers’ intelligence,” IEEE Trans. Knowl. Data Eng., vol. 25, no. 1, pp. 220–232, Jan. 2013.

[4] P.-Y. Chen, J.-W. Liu, and W.-T. Chen, “A fuel-saving and pollutionreducing dynamic taxi-sharing protocol in VANETs,” in Proc. 72nd IEEE VTC-Fall, 2010, pp. 1–5.

[5] P. d’Orey, R. Fernandes, and M. Ferreira, “Empirical evaluation of a dynamic and distributed taxi-sharing system,” in Proc. 15th IEEE Int. Conf. ITSC, Sep. 2012, pp. 140–146.

[6] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong. A cost-effective recommender system for taxi drivers. In 20th ACM SIGKDD International Conference on KDD, pages 45–54, 2014.

[7] Y. Huang and J. W. Powell. Detecting regions of disequilibrium in taxi services under uncertainty. In Proceedings of the 20th International Conference on Advances in Geographic Information Systems, number 10, pages 139–148, 2012.

[8] J. W. Powell, Y. Huang, F. Bastani, and M. Ji. Towards reducing taxicab cruising time using spatio-temporal profitability maps. In Proceedings of the 12th International Conference on Advances in Spatial and Temporal Databases, number 19, pages 242–260, 2011