Mindtree
A Larsen & Toubro Group Company

OVERCOMING PRACTICAL ENTERPRISE CHALLENGES FOR DEEP LEARNING

Best Practices to adopt and deliver Deep Learning solutions for enterprises

A Mindtree Whitepaper | April 2020

Abstract

We hear, see or experience AI and Machine Learning in our everyday life these days. Autonomous cars from Google, face or voice recognition to unlock your phone, or that amusing app that tells you if your friend is happy or sad when you click his or her picture. AI has been on the forefront of the technology revolution for the last half a decade. The recent developments in research and technology in this field has given the necessary spark for enterprises to start applying AI and advanced machine-learning solutions. However, practical business problems that leading organizations are trying to solve with AI are a lot more complex than classifying an image as a cat or a dog, and with that complexity comes its own set of problems. Implementing a Deep Learning technology (de facto core of AI techniques) has many technical and functional requirements that are hard to meet when operating in the real world with businesses. In this paper, we explore the variety of challenges that a data science team can face while implementing an AI solution at the enterprise level and reveal some of the best practices to efficiently deliver and adopt AI solutions in an organization.

Executive Summary

WHAT DEEP LEARNING SOLVE?

Business problems that require to generate insights by making predictions or mine patterns from unstructured and unconventional data such as images, text and voice can be solved with Deep Learning. Traditional Machine Learning solutions generally fall behind in terms of accuracy and performance when they deal with such cases. **Business use cases ranging from brand sentiment, manufacturing efficiency and process optimization can all benefit from Deep Learning solutions.**

WHAT ARE THE PERTINENT ROADBLOCKS FOR A DEPLOYING DEEP LEARNING SOLUTION?

Enterprises operate under many constraints when working with any technology. The key success factors for any Deep Learning solution is availability of large labeled data sets, knowledge of state-of-the-art techniques in building Deep Learning models and understanding innovative frameworks for deployment and validation of the solution. Meeting these requirements are often hurdled by:

- Lack of good quality and large volumes of training data relevant to the business use case
- Failure of building robust models with high accuracy due to lack of data
- Poor deployment architecture and higher costs per insights since Deep Learning demands higher computational resources
- Black box characteristics of Deep Learning models lead to difficulties in validating outputs through traditional business feedback

WHAT IS THE MANTRA FOR SUCCESSFUL PROOF OF CONCEPTS FOR DEEP LEARNING SOLUTIONS?

Tackling each challenge with the best practices can help ease the adoption and validation of Deep Learning solutions. An integral part of success is to have the know-how of state-of-the-art Deep Learning modeling techniques and Big Data architectures. Systematically tackling the identified challenges involve:

- Clearly defining the business problem and breaking it into simpler mathematical problems that can be solved through Deep Learning models
- Understand data limitations and build a model training strategy accordingly
- Employ the advanced techniques of Deep Learning modeling such as data augmentation and transfer the learning to improve and optimize accuracy of models
- Deploy models with cloud-enabled solutions to reduce fixed costs and operationalize the solution at reduced costs per insights
- Validate the model with the business considering the continuous learning and evolution of the model through a feedback mechanism and considering all relevant business KPIs such as cost of incorrect predictions, benefits/upside and opportunity costs

REAL WORLD AI USE CASES

At its core, Deep Learning is an advanced version of Machine Learning that can deal with a variety of unstructured data without the need for explicit pattern mining or feature engineering and selection. This makes it significantly more useful in today's age, where businesses often find themselves dealing with data like images and text, from which manually analyzing and mining information is difficult, time-consuming and sometimes next to impossible. A few leading examples that we see where Deep Learning is enabling business solutions and creating value are:

- **Image and text analysis for gauging consumer behavior towards brand and marketing campaigns.** Say, a consumer product company wants to understand the sentiments of its customers with respect to their brand and customer service. It collects valuable data in the form of tweets, social media posts and blogs, which contain the direct voice of customer. The images of the brand elements (product, banners etc.) the customers are uploading are signals of what people are talking about with respect to your brand. One can correlate the sentiment in the social media posts and correlate that to existing social media marketing campaigns and spends. It helps the company track what kind of brand images are going viral, or if there a spurge of 'unhealthy' brand images like crushed product containers, negative comments and so on. By leveraging Deep Learning for images and text, the organization can carefully monitor and score such customer activities. The value generated goes even further by helping maintaining a strong brand sentiment and optimizing marketing expenses.

A key value addition of using a Deep Learning solution is that it automates the process of detecting the contents and category of images that are posted by customers, a task which when done manually, will require additional human resources, time and effort. We will talk about this use case in further detail during the latter part of this paper where we dive into the specifics of how a consumer beverage company utilizes Deep Learning for computer vision to help classify the customer's social media posts according to the brand element (beverage containers).

- **Identifying manufacturing defects using Internet of Things (IoT) and Deep Learning.** Manufacturers spend significant time in detecting defected goods delivered by their assembly line. It also incurs significant costs if they are unable to remove defected goods before distribution. With the advent of IoT and sensor/camera technologies, manufacturing facilities are equipped with the tools to capture images and features of the assembly line processes and the manufacturing work in progress goods. However, traditional software tools are mostly rule-based and cannot adapt to the changing nature of the manufacturing defects and are limited in their predictive capabilities. In such a case, a robust image recognition algorithm based on Deep Learning can run through the image feed or even video footage from drones / facility cameras, and then identify and flag defective pieces in real-time, saving efforts downstream in loading and distribution. The solution can bring down lead times in supply chain and optimize the defect rate.

- **Image analysis for predicting image attributes.** Online travel and tourism aggregators collect images on the properties they host, but do not leverage the same for audits or analysis. A key requirement of the revenue model is to accurately set the prices of the aggregated properties (hotels, bnb's etc.). To improve this process, Deep Learning solutions are being used to predict prices/ ratings of a hospitality property on their website by analyzing the uploaded pictures and not asking for any additional information from the vendor. Using Deep Learning algorithms, which couples image feature recognition along with price prediction models, vendors are efficiently optimizing the pricing of their products and at the same time reducing manual intervention and lead times to hosting new portfolios. The solution further helps in preventing price anomalies and increase customer satisfaction.

These use cases have probably given a glimpse of the myriad of Deep Learning solutions possible for pertinent business problems. However, Deep Learning compared to other technologies, is still in a nascent stage. Most solutions implemented have to go through a rigorous process of experimentation and research in order to deliver the final product. At an enterprise level, there are several constraints and challenges when dealing with implementing Deep Learning solutions. Thus, it is difficult for data science teams to execute successful proof of concepts. Keeping aware of best practices and state-of-theart methods in Deep Learning can help minimize the challenges faced, as well as improve the performance of AI solutions.

CHALLENGES IN IMPLEMENTING DEEP LEARNING SOLUTIONS

In an enterprise, pertinent challenges for implementing a Deep Learning solution are around the volume of data and matching business expectations of a predictive/analytical system. AI is popular because of the accuracy that Deep Learning model promises when dealing with unstructured data and the large number of research use cases delivered by AI vendors like Microsoft, Amazon, IBM and Google. However, the real-world use cases are different in terms of how the problems are defined and the surrounding constraints, and lack the flexibility and support (time and effort) that these well-known solutions were privileged to have. From the conceptualization of the business problem to the final deployment of the solution, we can track key challenges faced by a data science team across the development pipeline.

- **Lack of data corpus** – Deep Learning models evolved under the paradigm – 'given the right amount of data, the model can identify and predict underlying patterns.' State-of-theart Deep Learning models, which predict with near to 100% accuracy, have been trained on millions of labelled images. The famous cats and dogs prediction problem uses around 50000 images of each (cats and dogs) to train a Deep Learning architecture which can predict an image of an animal as a cat or a dog with ~99% accuracy. In most scenarios, the data collected is of a much lower order. Moreover, getting accurately labeled unstructured data is even more difficult. Businesses often need to label unstructured data manually to facilitate training of Deep Learning models, a time-consuming and expensive task. In most scenarios, businesses can provide data volumes in order of hundreds or thousands. Building high-accuracy models with small data sets is one of the biggest challenges of Deep Learning.

- **Unavailability of flexible ready-made solutions** – Microsoft has a spectrum of Deep Learning applications delivered through APIs, and so do AWS (Amazon), IBM and Google (offered through APIs and other cloud services). These cover a good range of use cases from image analysis and object detection to more complex ones such as topic generation and text summarization. However, to a great extent, these are generic models trained on large data sets of widely available public data sources. To enable these solutions in the business context requires higher level of customization, which is not possible. Consider a beverage brand that deals with poor quality images captured from smartphones and uploaded on social media pages. Applying a generic image classification model can help detect categories of the image to an extent, but it will not be able to differentiate between damaged (unhealthy) beverage containers vis-à-vis healthy ones. These cannot be broken down to specific parts of the model as the available services are not flexible enough to extract sub-modules.

- **Ambiguity of project lifecycle** – Any data science problem, no matter how generic, can never be a complete off-the-shelf solution. Designing an accurate Deep Learning model is an iterative process and has dependencies on the volumes of data available, quality of the data and the complexity of the scientific problem (classification, regression etc.) that needs to be solved. Because of these characteristics, it is very difficult to plan project lifecycles transparently and accurately. Setting milestones and deadlines are equally challenging and the amount of effort needed from the data science team can only be estimated once there is full clarity on the data quality and availability, as well as the pre-processing requirements before moving to the modeling phase. In such scenarios, businesses face challenges to plan and estimate timelines and efforts for Deep Learning projects, and are hesitant to undertake complex Deep Learning solutions without prior experience

- **Demanding technology requirements** – Deep Learning algorithms are powerful techniques, but at the same time require higher computing resources. To set an understanding of this in the simplest way, a Deep Learning model has ~1000 times the computational requirements of linear regression model (which is practically 1 algebraic/mathematical equation). And, this factor increases exponentially as the complexity of the deep neural network architecture increases.

Implementing a deep neural network thus requires the latest cloud computational services, and the deployment architecture has to be carefully planned in order to optimize the utilization of resources. It often becomes hard to realize the best way to productionize a Deep Learning model. Compared to traditional descriptive analytics or simpler analytics solutions like market mix models, Deep Learning solutions demand a greater cost per insight. Inefficient architecture or misaligned deployment frameworks lead to higher operational costs of the solution (low ROI) and an unfulfilling solution (low utilization by business users), which eventually lead to slower or hindered adoption

- **Treating Deep Learning as a one-time solution** – Traditional technology solutions are mostly a one-time resolution to a pressing business problem. For example, once the organization decides to move its infrastructure to cloud, it undertakes the migration and that ends the project. Iteratively, there are only minor evolutionary phases such as migrating other smaller modules, newer data sources etc. Deep Learning works differently. The first solution is often the stepping-stone, while the model and architecture has to continuously evolve. The model's accuracy needs to be continuously improved by re-training it on new data. The architecture also needs to be scalable to adopt increasing volumes. Insights generated need to be regularly monitored to realize continuous value. In such cases, defining the long term technical road map is important and decisions on how we keep collecting new data points, re-training the model, and automating the collection and labeling of data are often overlooked during the planning and design phase, leading to road-blocks in the future.

- **Ambitious and ambiguous expectations from black box models** – Deep Learning frameworks can be considered to an extent as a black box model i.e. the exact workings of the model is hard to replicate manually or visualize. The internal workings of the model are abstracted and the focus of the solution design is mostly limited to developing the architecture of the neural network and deciding on the hyper-parameters (parameters defining the structure and constraints of the model). Deep Learning models excel at identifying patterns and making predictions given good training data. The black box characteristic often hinders its adoption in many business scenarios. Interpreting results are limited with Deep Learning compared to other models like decision trees and linear regression. Deep Learning models need to be validated with real data and well-designed experiments, and a level of trust on the model has to be built empirically (especially because it surpasses traditional models in its predictive capabilities). Published research on Deep Learning models claim to achieve close to human predictive accuracy when it deals with use cases such as predicting the category of an image or the sentiment of a text (which humans can achieve with almost 100% accuracy). This leads to businesses developing high expectations from Deep Learning models. For any solution using Deep Learning, especially for cases where proven human resource's capabilities are close to 100% accuracy, the expectation is that a Deep Learning solution will provide the same correctness. Theoretically, this assumption is true, but experimentally achieving it requires surpassing many challenges.

MINDTREE'S APPROACH FOR DESIGNING A DEEP LEARNING SOLUTION

Since we are now clear about the biggest challenges that implementation of a Deep Learning solution can encounter, we embark upon a more fruitful journey of delivering a Deep Learning solution. To elaborate on the best practices, we will show the approach taken to solve a use case mentioned earlier. The objective is to develop a solution that can help a consumer beverage company identify the nature of the social media images that are uploaded by its customers, correlate that to its ongoing social media campaigns and build other sentiment analysis solutions on top.

- **Breaking down the larger business objective into a simpler Deep Learning problem statement:** The key objective is classifying images uploaded by customers on social media as a positive or negative image. A positive image can be one which has elements of the brand such as the beverage container (bottles, cans etc.) in a 'healthy' state (undamaged, shown in positive mood) and a negative image can be one where the same elements are present in an 'unhealthy' state – crushed, damaged or abused.

- **Understanding availability of data:** Since we are dealing with images here, it is clear that we will have to build a Deep Learning model since traditional Machine Learning algorithms will most probably not perform well. What we have is a set of images of two types of containers, bottles and cans. Thus, we have to deal with four categories of images- 'unhealthy' cans, 'healthy' cans, 'unhealthy' bottles, 'healthy' bottles. As expected, one of the challenges we can see upfront is that currently, the set of collected images is a very small sample – around 160 images per category. These images were manually labeled (using helper open-source tools) for the purpose of the solution. Additionally, adding labeled images from public data sets (internet sources) can be considered as one option, but it is essential that any image used to train and validate a AI model, is as close to the actual real image that the model will have to predict. Public data sets mostly have high quality and isolated images of similar containers whereas customers upload lesser quality images which contain more objects than just a can or a bottle. Moreover, the model needs to be trained specific to the brand's own bottle and cans (in shape and size). Thus, for the purpose of a proof of concept solution, we will restrict ourselves to using the only the available data.

- **Building a base model:** Using experimental approaches, we choose a base model which is a proven convolutional neural network (CNN) architecture to fulfill the objective here – to classify images as defective cans vs defective bottles. The architecture used is shown in exhibit 1 and 2.

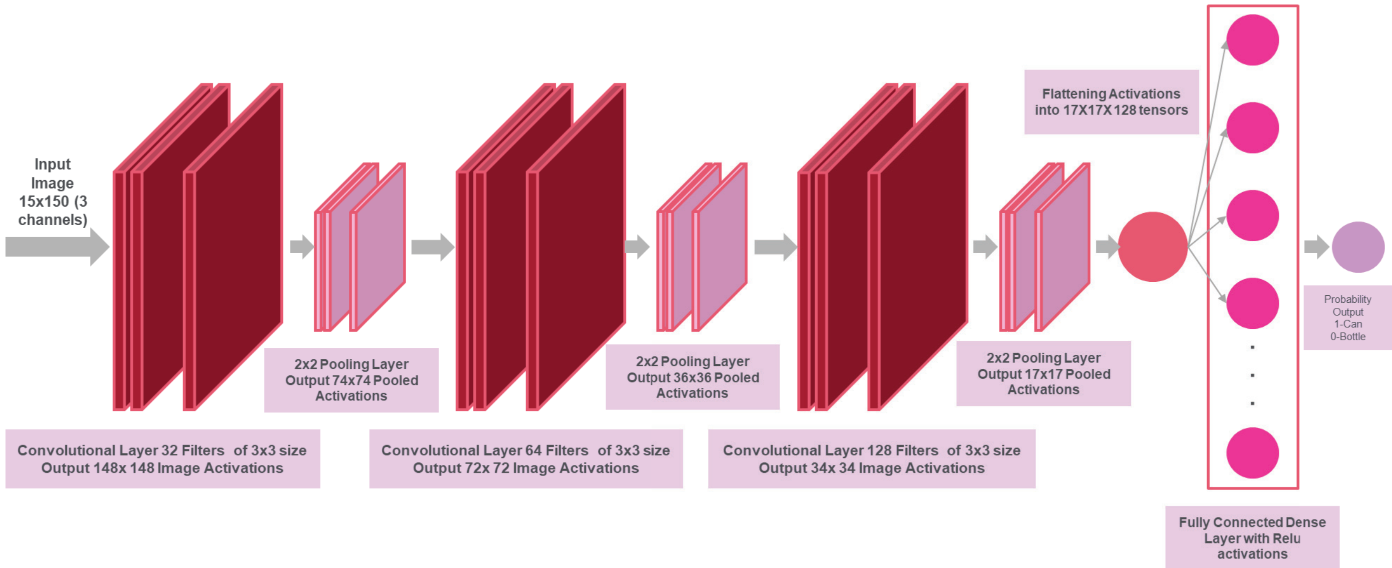

Exhibit 1

This neural network model will take images in the form of 150 x 150 pixels with three channels (RGB) and output a probability score of the image belonging to the categories (or classes) i.e. cans or bottles (defective and healthy). To understand the complexity of the network, it is important to note that this model trains ~19 million parameters, a task that is difficult, given the limitations we have in the number of train images. The training is performed by breaking the existing data set into 120 images for training, 20 images for validation (while training). Another 20 images are isolated for testing (unseen data during training process).

For evaluating this model, choosing 'accuracy' (number of correct predictions/ total number of predictions) is a good measure of evaluating our model. This base model, after training on the 240 training images, gave us an accuracy of 82%. What we further observed was that

- The training accuracy (accuracy we get on the training set) is ~98%, while for the validation set and test set is around 82%. Which means that the model is highly over-fitted to the training data and hence, the performance on 'unseen/untrained' data is poorer. This is as expected since we have just 240 train images (compared to 100,000 images that go into training state-of-the-art classifiers!)

- Accuracy of 82% is quite low (technically called training bias) when you compare it in the reference of the business use case. The business would want near 99% accuracy. (Assuming that a human mind can also make 1% error)

- The positive inference that we make from this model is that our validation and test accuracy are in a similar range (82% vs 85%), thus implying that our model is robust enough (technically called model variance).

Technically, this model is a good start. To solve the problem of overfitting, we can apply best practice techniques like regularization (adding dropout layers, lasso etc.) which help the training process to generalize the model better. However, to improve our accuracy of the model itself, we will explore other state-of-the-art techniques and best practices.

Improving the accuracy of the model - The key challenge with increasing the accuracy in a Deep Learning model is that we do not have sufficient labelled training data. This is often a halting point for most predictive modeling experiments. There are however, a number of tactics specific to Deep Learning that should be used to help improve the model. One of the best practices to begin with is to try **data augmentation.** It is similar to the 'over-sampling' technique often used in Machine Learning to increase the proportion of training data for a particular class. Essentially, what it does it randomly creates more samples of data from the existing training data within a range of mutations that it can do to the training data. In this context, imagine taking the 240 images of cans and bottles and mutating them by doing one of the following:

- Zooming in for a small percentage
- Rotating the image slightly
- Changing a few small pixels randomly
- Compressing the image in width or height

This ensures that the images remain faithful to the class we are training for - very similar to the other images in the training data, but technically, the images are different. We can create such random samples to technically increase the size of the training data and improve the model training process. We however have to ensure that our training and validation (and test) data are of similar distributions. Otherwise, this method can significantly over-fit the model to the training data (which is why it was not prudent to use generic public data set images for training as stated earlier). The architecture of the deep neural network does not change; only the way the input is fed into the training process is altered to enable this process. When we experiment this with our current data set, we again find some improvement on our accuracy.

Another very important and useful technique in Deep Learning is transfer learning. Conceptually, transfer learning translates to using an existing trained model (both the architecture and the trained parameters) and re-using it for the given problem. This is possible as in Deep Learning, a part of the neural network works towards training itself to capture generic features. In the context of image classification (our current problem), this means learning how to detect edges, curves, shapes, depth etc. considering these pre-trained models have been trained on millions of images, it has good accuracy on new images. As a best practice, it is now expected that most Deep Learning problems should apply transfer learning to the best possible extent to leverage existing solutions and research.

In our architecture, the convolutional base (convolutional layers and pooling layers) are part of the architecture that help in capturing image features, while the subsequent layers capture information about the category to which it has to classify (bottles vs cans). Thus, we re-build our architecture replacing the convolutional base with an existing pre-trained model. For our problem statement, we chose an architecture called **InceptionNet from Google.**

Using these two techniques, first in isolation and then in combination, we were able to test out their impact on the model accuracy. The base model gave a test accuracy of 82%, which after combining the two techniques, was boosted to ~98%, which is very close to the earlier mentioned human threshold (99%), and thus will be more beneficial and acceptable in the business context. The incremental results from using these techniques to improve the model can be seen in exhibit 4.

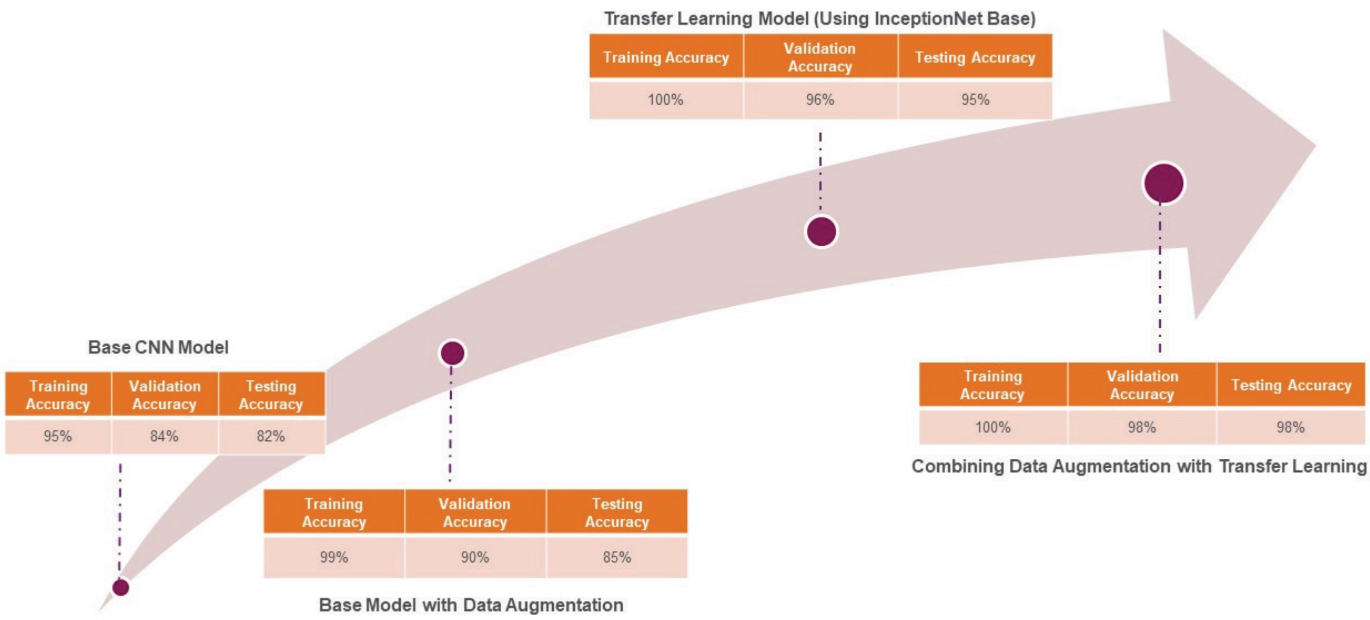

Exhibit 4

Deploying a robust architecture - Deep Learning typically requires high computational resources for training. Most enterprises productionize such solutions by deploying it on cloud, using services from the likes of Microsoft, AWS or Google. Since these services are pay per use, the risk of low utilization for higher cost is minimized. In order to make optimum usage of resources while maintaining lower costs of services, the design of the solution must be carefully evaluated.

The final insights did not demand a real-time prediction algorithm to predict images as soon as a customer uploads a picture. Instead, the purpose is to periodically collect social media images and analyze them to see if they are being uploaded as bottles or cans and if they are 'healthy' or 'unhealthy' for the brand. The bigger picture is also about incorporating data from social media marketing spends and build correlation and other analysis on top of the predictions. Thus, we explored a design suitable for a batch processing ecosystem.

To reduce operational costs, we opted for cloud-enabled AI/ML services from Microsoft. Considering how a Deep Learning model learns, we first set a threshold of new data points that should be collected before re-training. Setting up architectures in which the model iteratively re-trains without actually improving its learning and accuracy due to lack of sufficient new data points is one of the key reasons for higher production and operational costs. In our experiments during the modeling phase, we were able to see marginal differences on extending the data size by 30-50 images. This was even after we started with a small data set (240 training images) to begin with. Thus, we modeled our Machine Learning flow such that the model re-trains only after 50% increase in its data size. Another empirically proven technique is to train the model frequently (10% increase in data size) for the first 3-5 iterations and then reduce the frequency (50-10% increase in data size). Since we already have 98% accuracy, maintaining a low frequency is a good option to begin with. This helps in optimizing the run-time costs of the production architecture, thus adding more value to the business and reducing the cost per insights.

Another important consideration that Deep Learning models face is compatibility with different deployment environments and frameworks. Technological advances in the space of open source frameworks and support from leading cloud vendors has removed this challenge to great extent. As an AI/ML solution however, we ensure that we deployed using best inter-compatible frameworks. We implemented the solution in Keras with Tensorflow backend, which is considered the de-facto framework for Deep Learning. For deploying in the Azure ecosystem, we serialized the Keras implementation into ONNX format or Pickle format (a compatible neural network serialization format).

The logical architecture deployed is shown in exhibit 5.

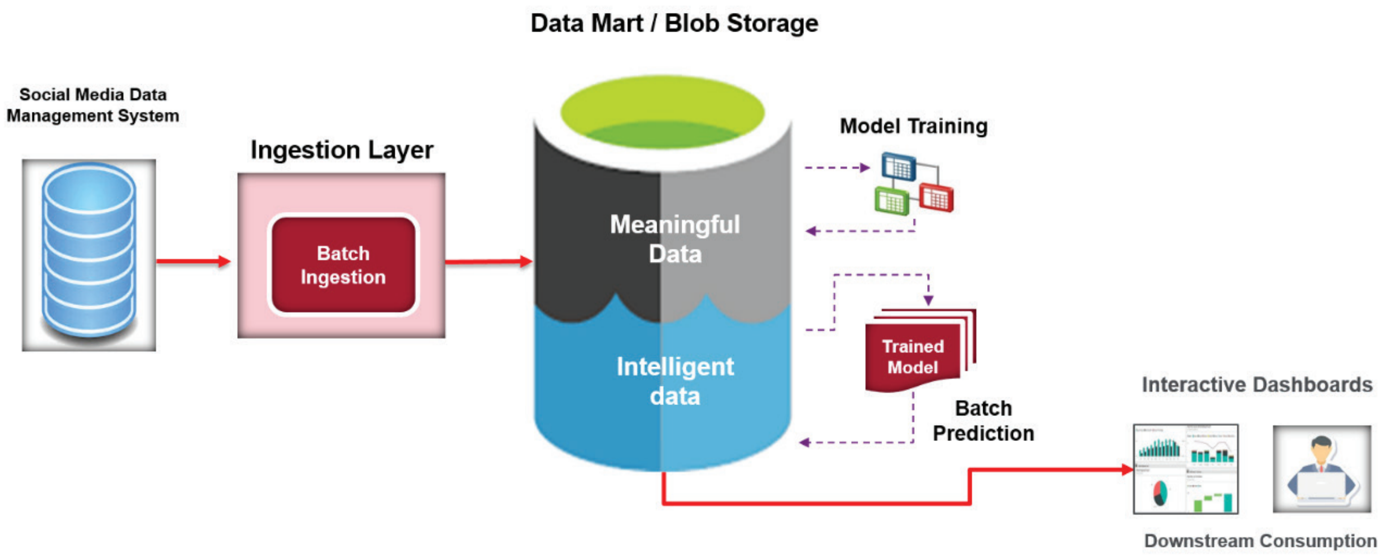

Exhibit 5

The architecture ensured that we could periodically re-train the model as we get more images, which is critical for the continuous learning and evolution of a predictive analytics solution. It also enabled interactive dashboards to be built on the intelligent data (predictions and other insights). Moreover, deploying this through cloud services ensured that we only incur computational costs when the model is running (during training and batch prediction) and for storage of images, essentially bringing down the fixed cost of the solution to the absolute minimum possible.

Maintaining the continuous learning cycle – Deploying a trained model is only half of the job for a Deep Learning (or any Machine Learning) project. It is essential to ensure that there is a pipeline for continuously evolving the solution. A critical part of ensuring that is to have a feedback system in place. The model keeps making its predictions on new images being uploaded on social media. These predictions are shared as insights to business users. A small component of the downstream consumption (such as the dashboards etc.) should also facilitate feedback from the business users on incorrect predictions and other analysis outputs.

This mechanism ensures that continuous training of the model does not involve additional costs for labeling data. In the initial iterations of our use case, we were dealing with approximately 160 images per category (cans and bottles).Manually labeling them was the only way due to the small size of the data set. Using open source tools for labeling (which requires manual sorting but reduces human effort in cropping the images and setting the labels). This was not a huge cost in terms of time and effort. But, incrementally labeling data becomes a cost head for the solution and also requires timely manual intervention. By incorporating a feedback mechanism on the predictions, the process gets streamlined.

Evaluating benefits - Using Deep Learning in enterprises for solving business problems is still a very nascent technology. Marketing, sales and operations heads of leading industries are yet to pick up the nuances of this technology. In the context of business, the key success parameters are agility, accuracy and cost. Deep Learning models are indeed more costly since they require more computing resources, and the build time (training and offline evaluation) is higher than traditional statistical models. However, they provide much greater accuracy than most traditional approaches. Let us take our use case where we have been able to achieve a model with the following accuracy metrics:

- Train Accuracy - ~100% (low volumes of training data, over fitted to training data, will predict very similar images with almost 100% accuracy)
- Validation Accuracy – 98% (chances of 2% error when predicting new images)
- Test Accuracy – 98% (model is robust enough that the error rate will be not be volatile for new images)

Understanding why prediction accuracy is at a certain level or which drivers are leading to a false prediction is difficult in the case of Deep Learning models. Thus, building risk averse (against false predictions) business rules over the model outputs a weak viable option. The alternative is to build a holistic evaluation around the model's output and performance. The model accuracy is good, but can it justify the cost is what needs to be carefully evaluated to make a final judgement on the ROI of the solution.

To dive deeper, we should also understand the concept of 'Bayes' error. The theory suggests that as time (indicator of training data size and computational resources) increases, a Machine Learning model (or Deep Learning) can beat the accuracy of human predictions (in our case of predicting images as cans or bottles, it can be assumed to be ~99%) and will eventually reach a stagnation point. This stagnation point is called Bayes optimal error and technically, it is impossible to achieve a better accuracy than this point.

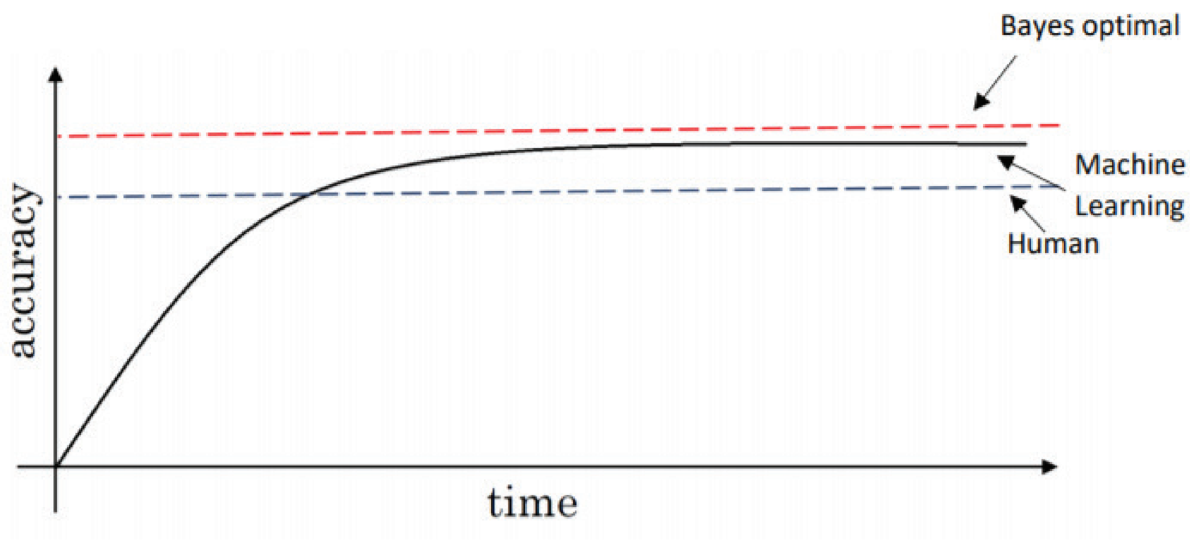

Exhibit 6

In our case, the standing accuracy is not Bayes optimal - the primary reason being, we are heavily limited in the amount of training data – the common business constraint for Deep Learning. However, a sub-optimal accuracy can also be valuable to the business.

The need is to create a validation framework which evaluates the output of the model considering all business and model-related statistical factors. Some of the key factors we considered for our use case were:

- The cost of running predictions on new batches of images
- The accuracy of the prediction (in our case, it was initially at 98%)
- The uplift or business benefit derived from a correct prediction (98% chance of correct predictions)
- The loss or cost of incorrectly predicting an image (2% chances of incorrect predictions)
- The opportunity cost. In this particular scenario, the opportunity was to utilize Deep Learning to effectively track brand messaging and marketing. A traditional alternative would be to have a human resource perform the predictions manually .

By having a good feedback mechanism in place, we can monitor the model outputs when in production, and keep inferring the results against the mentioned KPIs. As we get more data, the model accuracy reaches closer to Bayes optimal level, which is the point at which we can clearly validate the productivity and cost effectiveness (ROI) of our solution.

BEST PRACTICES TO FOLLOW

The journey elaborated for our use case constitutes some of the best practices in the industry to implement Deep Learning solutions for relevant business use cases. While granular nuances may vary from case to case, the approach for problem identification, AI/ML modeling (training, build and selection) and deploying a positive ROI solution remains the same.

As seen, the key pain points are around the availability of data and the lack of deeper knowledge of how Deep Learning models work. As democratization of Deep Learning matures, these best practices will become pervasive and the adoption of such solutions will become first nature to businesses. The best practices elicited above will remain relevant and might change only in terms of the specific technologies in use or addition of newer techniques in line with the current practices. Our experience with Deep Learning has enabled us to outline our approach in a generic manner to ensure smooth implementation and adoption of such solutions for use cases as mentioned in this paper.

- Dig deep into the problem- Break down the business problem into smaller, more precise mathematical/Machine Learning complications
- Understand limitations of Deep Learning - Thoroughly understand the limitations of raw data and accordingly set goals for achieving prediction accuracy or business benefits
- Start simple and set the right success criteria - Build a practical baseline model to start with
- Iteratively improve towards the optimal solution - Build the best possible Deep Learning model using the state-of-the-art architectures and boost accuracy within the practical constraints of data by using advanced techniques like
    a. Data augmentation
    b. Transfer learning
    c. Regularization and optimized hyper parameter tuning

- Holistic evaluation and not just accuracy: Evaluate the models offline using metrics like accuracy and compare against baseline, human accuracy and Bayes optimal accuracy. Also evaluate the benefits of the solution in action. Perform a holistic evaluation of the solution in terms of measuring the business outcome (ROI) of having the Deep Learning solution vs as is

- Deploy smart – Deploy the solution by leveraging cloud services and streamline the solution pipeline to optimally reduce cost per insight

A Deep Learning solution is only as good as its design and training. The implementation of state-of-the-art techniques and proven best practices can ensure it gets to deliver the promises it shows.

Anirban Pramanik	**Amartya Sen**	**Rudrasatyapadma Jaddu**
Senior Data Scientist, Data Science and Engineering Group, Mindtree	Lead Data Analyst, Data Science and Engineering Group, Mindtree	Software Engineer, Big Data Analytics Group, Mindtree

About Mindtree