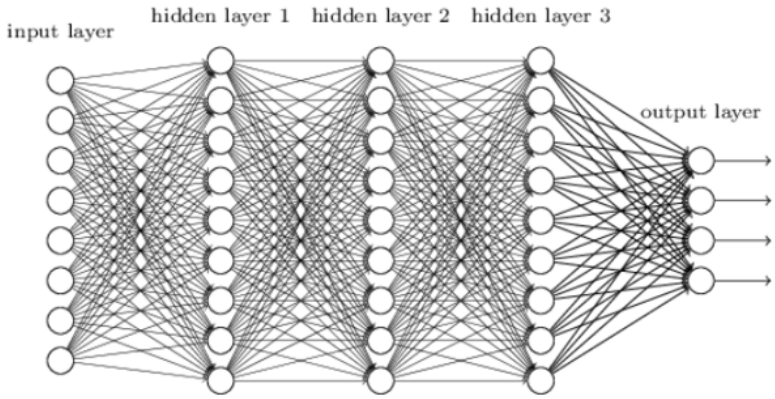
Deep Neural Network Implementation Issues

Deep neural network (DNN) is an artificial neural network (ANN) which has multiple layers between the input and output layers. Deep learning has made speech recognition accurate enough to be used in carefully controlled environments. Voice recognition systems such as Cortana and Siri use deep neural networks to imitate human interactions. Overtime, these apps learn the nuances and semantics of our language. For instance, Siri can identify her trigger phrase 'Hey Siri' under any conditions. Using this, you can access function commands like ordering a pizza, setting alarms, calling someone, opening files, and making reservations at your favorite restaurant.

The following are the implementation issues associated with the DNN model:

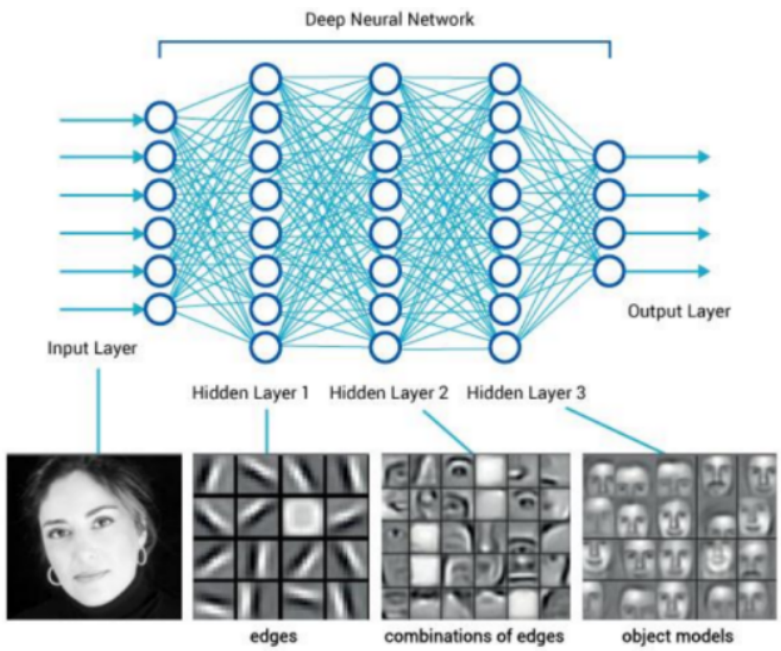
- DNNs are prone to overfitting due to the presence of added layers of abstraction, which allow them to model rare dependencies in the training data. For instance, the Google API uses densely connected layers. These models may have great learning capacity, but they are also prone to overfitting. This makes it difficult to learn from features with local correlations.
- DNNs require a large amount of data to train them. DNNs, unlike other neural networks for which feature information is provided within the input, require enough data to identify features on their own. For instance, voice recognition systems like Alexa, Siri, and Cortana will have to learn from large amounts of training data to obtain clear signals from noisy speech. They will also have to train over a large vocabulary of words to understand commands.



The figure above shows the components of the DNN: input layer, hidden layer/ layers, and output layer

- DNNs are known as the "Big Black Box" since they are unable to provide a suitable explanation for how the model arrived at the solution. For instance, if a system identifies a suspicious area in a medical image, the radiologist may not be able to see the specific detection criteria, since DNNs are limited by how these tools identify features and infer details.
- DNNs need to take into account several training parameters, such as number of layers, number of units per layer, initial weights, and learning rate. Sweeping through the parameter space for parameters that are optimal (modifying the value of hyperparameters by even a small amount can cause a large change in the model's performance) is computationally complex and time consuming.
- Relying on the default parameters and not performing hyperparameter optimization can negatively impact the model's performance. An experiment was conducted to see the

performance of a voice recognition system with default parameters and with optimized parameters, and the results showed that the system with the optimized parameters achieved significantly better accuracy.



The figure above shows the learning workflow of the DNN model where the input of each layer is the output of the previous one

- Online Deep Learning (ODL) is the process of learning DNNs in an online setting. The biggest challenge is picking a proper model capacity before the start of the learning process online. If the model is too complex, the learning process will converge too slowly, thus losing the desired property of online learning, and if the model is too simple, the learning capacity will be too restricted, making it difficult to learn complex patterns (Sahoo, Pham, Lu, Hoi, 2014).
- Diminishing feature reuse is a problem DNNs face. This is an

issue in which, many useful features are lost during the feedforward stage of the prediction. This is especially critical for online learning to avoid poor performance during initial training (Sahoo, Pham, Lu, Hoi, 2014).

- The rapid increase in the number of saddle points, and not local minima, poses a challenge to DNNs. These saddle points are surrounded by high error plateaus that can greatly slow down the learning process and give the illusion that local minima exist (Dauphin, Pascanu, Gulcehre, Cho, Ganguli, Bengio, 2014).
- Internal covariate shift during training is a problem that DNNs face. During training, the distribution of each layer's inputs changes, since the parameters of the previous layers change. This slows down the training since it requires lower learning rates and careful parameter initialization and makes it hard to train models with saturating nonlinearities (Loffe, Szegedy, 2015).

Sahoo, D. Pham, Q. Lu, J. Hoi, S. Online Deep Learning: Learning Deep Neural Networks on the Fly. 2014. Retrieved from: https://arxiv.org

Dauphin, Y. Pascanu, R. Gulcehre, C. Cho, K. Ganguli, S. Bengio, Y. Identifying and Attacking the Saddle Point Problem in High-dimensional Non-convex Optimization. 2014. Retrieved from: https://arxiv.org

Loffe, S. Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2015. Retrieved from: https://arxiv.org