

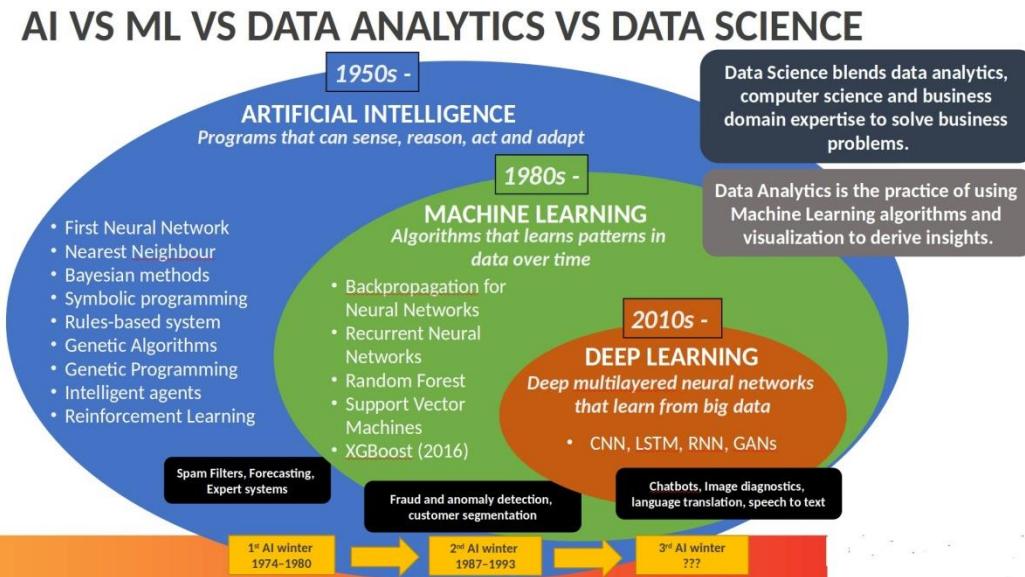
Data Science Interview Questions

(30 days of Interview Preparation)



Q1. What is the difference between AI, Data Science, ML, and DL?

Ans 1 :



Artificial Intelligence: AI is purely math and scientific exercise, but when it became computational, it started to solve human problems formalized into a subset of computer science. Artificial intelligence has changed the original computational statistics paradigm to the modern idea that machines could mimic actual human capabilities, such as decision making and performing more “human” tasks. Modern AI into two categories

1. General AI - Planning, decision making, identifying objects, recognizing sounds, social & business transactions
2. Applied AI - driverless/ Autonomous car or machine smartly trade stocks

Machine Learning: Instead of engineers “teaching” or programming computers to have what they need to carry out tasks, that perhaps computers could teach themselves – learn something without being explicitly programmed to do so. ML is a form of AI where based on more data, and they can change actions and response, which will make more efficient, adaptable and scalable. e.g., navigation apps and recommendation engines. Classified into:-

1. Supervised
2. Unsupervised
3. Reinforcement learning

Data Science: Data science has many tools, techniques, and algorithms called from these fields, plus others –to handle big data

The goal of data science, somewhat similar to machine learning, is to make accurate predictions and to automate and perform transactions in real-time, such as purchasing internet traffic or automatically generating content.

INEURON.AI

Data science relies less on math and coding and more on data and building new systems to process the data. Relying on the fields of data integration, distributed architecture, automated machine learning, data visualization, data engineering, and automated data-driven decisions, data science can cover an entire spectrum of data processing, not only the algorithms or statistics related to data.

Deep Learning: It is a technique for implementing ML.

ML provides the desired output from a given input, but DL reads the input and applies it to another data. In ML, we can easily classify the flower based upon the features. Suppose you want a machine to look at an image and determine what it represents to the human eye, whether a face, flower, landscape, truck, building, etc.

Machine learning is not sufficient for this task because machine learning can only produce an output from a data set – whether according to a known algorithm or based on the inherent structure of the data. You might be able to use machine learning to determine whether an image was of an “X” – a flower, say – and it would learn and get more accurate. But that output is binary (yes/no) and is dependent on the algorithm, not the data. In the image recognition case, the outcome is not binary and not dependent on the algorithm.

The neural network performs MICRO calculations with computational on many layers. Neural networks also support weighting data for ‘confidence’. These results in a probabilistic system, vs. deterministic, and can handle tasks that we think of as requiring more ‘human-like’ judgment.

Q2. What is the difference between Supervised learning, Unsupervised learning and Reinforcement learning?

Ans 2:

Machine Learning

Machine learning is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead.

Building a model by learning the patterns of historical data with some relationship between data to make a data-driven prediction.

Types of Machine Learning

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Supervised learning

In a supervised learning model, the algorithm learns on a labeled dataset, to generate reasonable predictions for the response to new data. (Forecasting outcome of new data)

- Regression
- Classification

Unsupervised learning

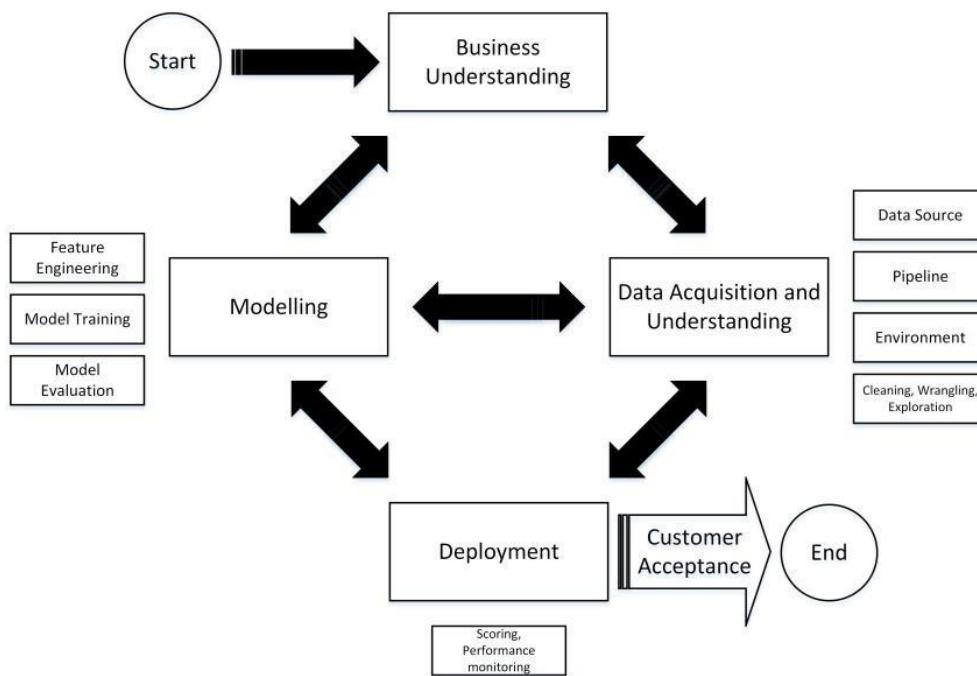
An unsupervised model, in contrast, provides unlabelled data that the algorithm tries to make sense of by extracting features, co-occurrence and underlying patterns on its own. We use unsupervised learning for

- Clustering
- Anomaly detection
- Association
- Autoencoders

Reinforcement Learning

Reinforcement learning is less supervised and depends on the learning agent in determining the output solutions by arriving at different possible ways to achieve the best possible solution.

Q3. Describe the general architecture of Machine learning.



Business understanding: Understand the given use case, and also, it's good to know more about the domain for which the use cases are built.

Data Acquisition and Understanding: Data gathering from different sources and understanding the data. Cleaning the data, handling the missing data if any, data wrangling, and EDA(Exploratory data analysis).

Modeling: *Feature Engineering* - scaling the data, feature selection - not all features are important. We use the backward elimination method, correlation factors, PCA and domain knowledge to select the features.

Model Training based on trial and error method or by experience, we select the algorithm and train with the selected features.

Model evaluation Accuracy of the model , confusion matrix and cross-validation.

If accuracy is not high, to achieve higher accuracy, we tune the model...either by changing the algorithm used or by feature selection or by gathering more data, etc.

Deployment - Once the model has good accuracy, we deploy the model either in the cloud or Rasberry py or any other place. Once we deploy, we monitor the performance of the model.if its good...we go live with the model or reiterate the all process until our model performance is good.

It's not done yet!!!

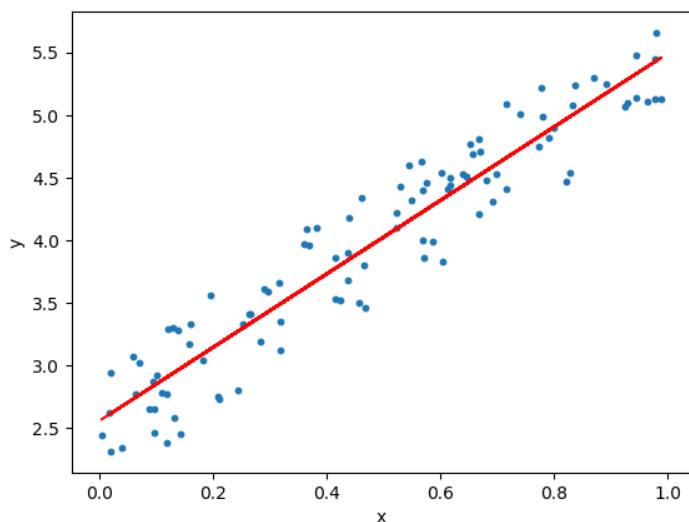
What if, after a few days, our model performs badly because of new data. In that case, we do all the process again by collecting new data and redeploy the model.

Q4. What is Linear Regression?

Ans 4:

Linear Regression tends to establish a relationship between a dependent variable(Y) and one or more independent variable(X) by finding the best fit of the straight line.

The equation for the Linear model is $Y = mX+c$, where m is the slope and c is the intercept



In the above diagram, the blue dots we see are the distribution of 'y' w.r.t 'x.' There is no straight line that runs through all the data points. So, the objective here is to fit the best fit of a straight line that will try to minimize the error between the expected and actual value.

Q5. OLS Stats Model (Ordinary Least Square)

Ans 5:

OLS is a stats model, which will help us in identifying the more significant features that can have an influence on the output. OLS model in python is executed as:

```
lm = smf.ols(formula = 'Sales ~ am+constant', data = data).fit() lm.conf_int() lm.summary()
```

And we get the output as below,

```
OLS Regression Results
=====
Dep. Variable: mpg R-squared: 0.360
Model: OLS Adj. R-squared: 0.338
Method: Least Squares F-statistic: 16.86
Date: Wed, 17 Jan 2018 Prob (F-statistic): 0.000285
Time: 14:07:51 Log-Likelihood: -95.242
No. Observations: 32 AIC: 194.5
Df Residuals: 30 BIC: 197.4
Df Model: 1
Covariance Type: nonrobust
=====
            coef  std err      t      P>|t|      [0.025  0.975]
-----
constant  17.1474  1.125    15.247  0.000    14.851  19.444
am        7.2449  1.764     4.106  0.000    3.642  10.848
=====
Omnibus: 0.480 Durbin-Watson: 1.065
Prob(Omnibus): 0.787 Jarque-Bera (JB): 0.589
Skew: 0.051 Prob(JB): 0.745
Kurtosis: 2.343 Cond. No. 2.46
=====
```

The higher the t-value for the feature, the more significant the feature is to the output variable. And also, the p-value plays a role in rejecting the Null hypothesis(Null hypothesis stating the features has zero significance on the target variable.). **If the p-value is less than 0.05(95% confidence interval) for a feature, then we can consider the feature to be significant.**

Q6. What is L1 Regularization (L1 = lasso) ?

Ans 6:

The main objective of creating a model(training data) is making sure it fits the data properly and reduce the loss. Sometimes the model that is trained which will fit the data but it may fail and give a poor performance during analyzing of data (test data). This leads to overfitting. Regularization came to overcome overfitting.

Lasso Regression (**Least Absolute Shrinkage and Selection Operator**) adds “Absolute value of magnitude” of coefficient, as penalty term to the loss function.

Lasso shrinks the less important feature's coefficient to zero; thus, removing some feature altogether. So, this works well for feature selection in case we have a huge number of features.

L1 Regularization

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M |W_j|$$

L2 Regularization

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M W_j^2$$

Loss function
Regularization Term

Methods like Cross-validation, Stepwise Regression are there to handle overfitting and perform feature selection work well with a small set of features. These techniques are good when we are dealing with a large set of features.

Along with shrinking coefficients, the **lasso performs feature selection**, as well. (Remember the 'selection' in the lasso full-form?) Because some of the coefficients become exactly zero, which is equivalent to the particular feature being excluded from the model.

Q7. L2 Regularization(L2 = Ridge Regression)

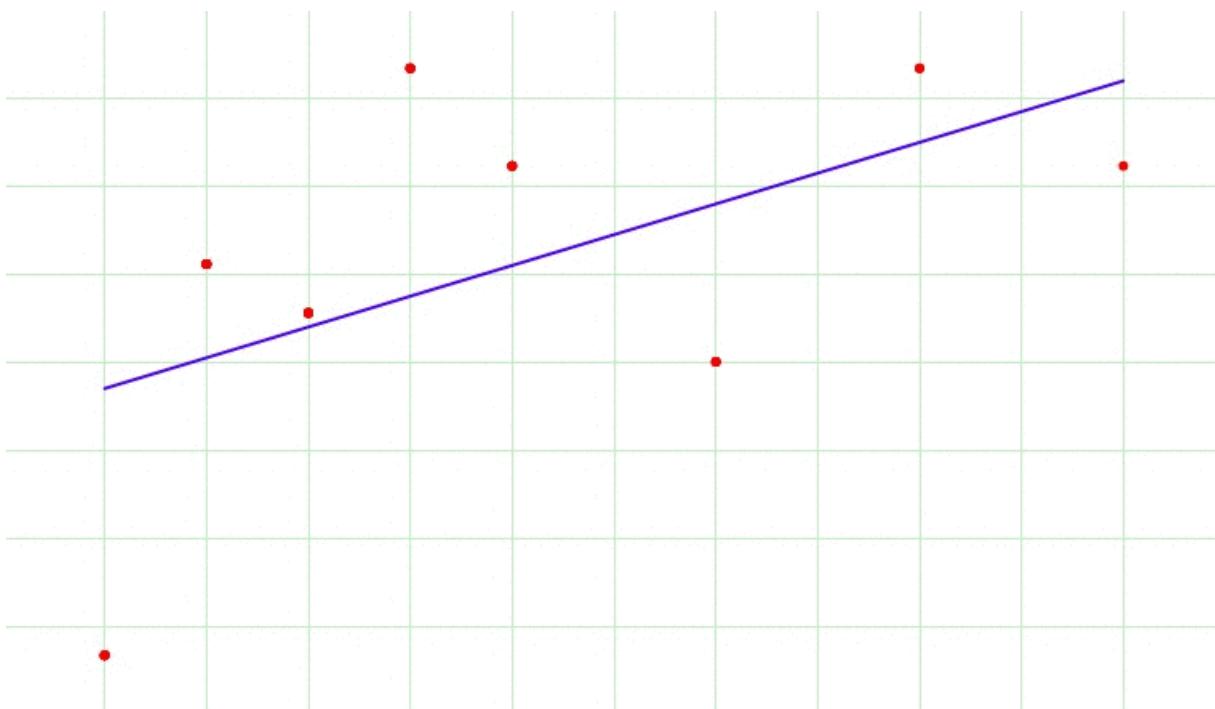
Ans 7:

$$\text{Cost function} = \text{Loss} + \frac{\lambda}{2m} * \sum \|w\|^2$$

Overfitting happens when the model learns signal as well as noise in the training data and wouldn't perform well on new/unseen data on which model wasn't trained on.

To avoid overfitting your model on training data like **cross-validation sampling, reducing the number of features, pruning, regularization**, etc.

So to avoid overfitting, we perform Regularization.



The Regression model that uses L2 regularization is called Ridge Regression.

The formula for Ridge Regression:-

$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2 \right]$$

$$\min_{\theta} J(\theta)$$

Regularization adds the penalty as model complexity increases. The regularization parameter (lambda) penalizes all the parameters except intercept so that the model generalizes the data and won't overfit.

Ridge regression adds "squared magnitude of the coefficient" as penalty term to the loss function. Here the box part in the above image represents the L2 regularization element/term.

$$\sum_{i=1}^n (y_i - \sum_{j=1}^p x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^p \beta_j^2$$

Lambda is a hyperparameter.

If lambda is zero, then it is equivalent to OLS. But if lambda is very large, then it will add too much weight, and it will lead to under-fitting.

Ridge regularization forces the weights to be small but does not make them zero and does not give the sparse solution.

Ridge is not robust to outliers as square terms blow up the error differences of the outliers, and the regularization term tries to fix it by penalizing the weights

Ridge regression performs better when all the input features influence the output, and all with weights are of roughly equal size.

L2 regularization can learn complex data patterns.

Q8. What is R square(where to use and where not)?

Ans 8.

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

The definition of R-squared is the percentage of the response variable variation that is explained by a linear model.

R-squared = Explained variation / Total variation

R-squared is always between 0 and 100%.

0% indicates that the model explains none of the variability of the response data around its mean.

100% indicates that the model explains all the variability of the response data around its mean.

In general, the higher the R-squared, the better the model fits your data.

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}}$$

Sum Squared Regression Error
↓
 $SS_{Regression}$
↑
Sum Squared Total Error

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

There is a problem with the R-Square. The problem arises when we ask this question to ourselves.** Is it good to help as many independent variables as possible?**

The answer is No because we understood that each independent variable should have a meaningful impact. But, even** if we add independent variables which are not meaningful**, will it improve R-Square value?

Yes, this is the basic problem with R-Square. How many junk independent variables or important independent variable or impactful independent variable you add to your model, the R-Squared value will always increase. It will never decrease with the addition of a newly independent variable, whether it could be an impactful, non-impactful, or bad variable, so we need another way to measure equivalent R-Square, which penalizes our model with any junk independent variable.

So, we calculate the **Adjusted R-Square** with a better adjustment in the formula of generic R-square.

$$R^2_{\text{adjusted}} = 1 - \frac{(1 - R^2)(N - 1)}{N - p - 1}$$

where

R^2 = sample R-square

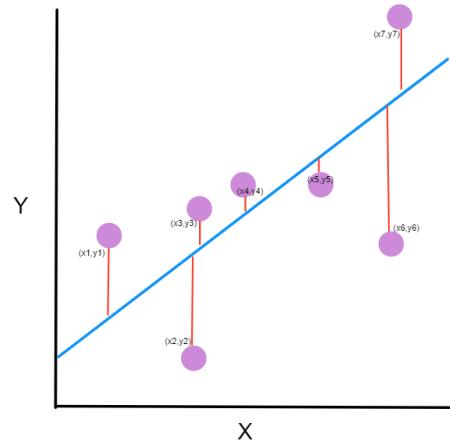
p = Number of predictors

N = Total sample size.

Q9. What is Mean Square Error?

The mean squared error tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line (these distances are the “errors”) and squaring them.

Giving an intuition



The line equation is $y = Mx + B$. We want to find **M (slope)** and **B (y-intercept)** that minimizes the squared error.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \tilde{y}_i)^2$$

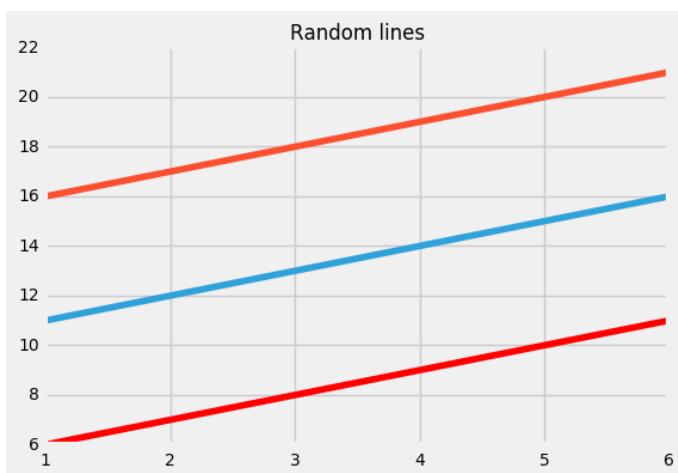
Q10. Why Support Vector Regression? Difference between SVR and a simple regression model?

Ans 10:

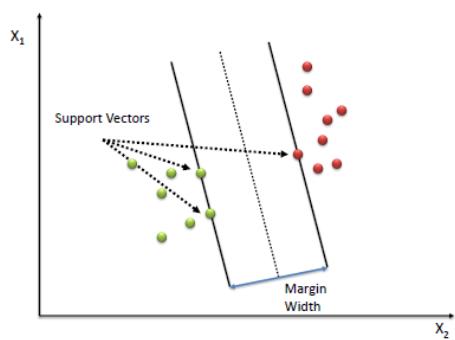
In simple linear regression, try to minimize the error rate. But in SVR, we try to fit the error within a certain threshold.

Main Concepts:-

1. Boundary
2. Kernel
3. Support Vector
4. Hyper Plane

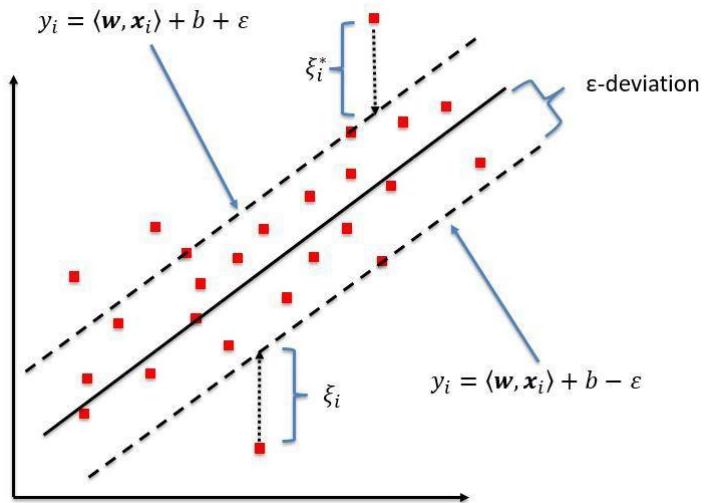


Blueline: Hyper Plane; Red Line: Boundary-Line



Our best fit line is the one where the hyperplane has the maximum number of points.

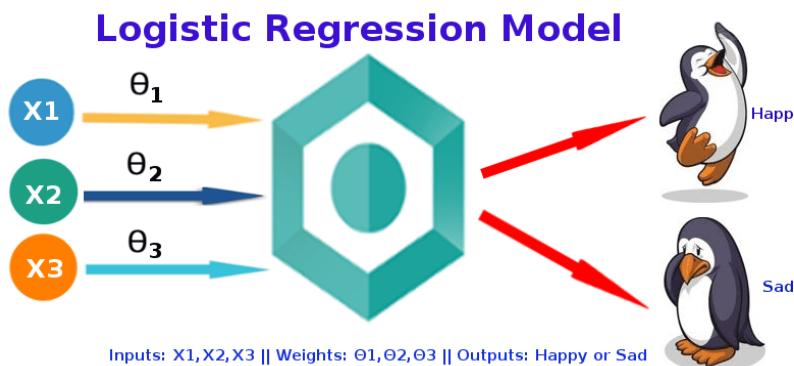
We are trying to do here is trying to decide a decision boundary at 'e' distance from the original hyperplane such that data points closest to the hyperplane or the support vectors are within that boundary line



Q1. What is Logistic Regression?

Answer:

The logistic regression technique involves the dependent variable, which can be represented in the binary (0 or 1, true or false, yes or no) values, which means that the outcome could only be in either one form of two. For example, it can be utilized when we need to find the probability of a successful or fail event.



Logistic Regression is used when the dependent variable (target) is categorical.

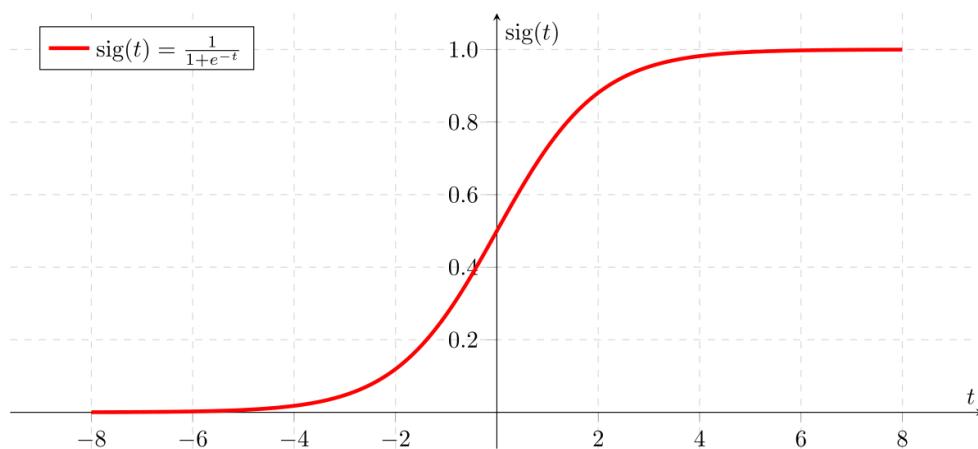
Model

Output = 0 or 1

$$Z = WX + B$$

$$h\Theta(x) = \text{sigmoid}(Z)$$

$$h\Theta(x) = \log(P(X) / 1 - P(X)) = WX + B$$



If 'Z' goes to infinity, Y(predicted) will become 1, and if 'Z' goes to negative infinity, Y(predicted) will become 0.

The output from the hypothesis is the estimated probability. This is used to infer how confident can predicted value be actual value when given an input X.

Cost Function

$$\text{Cost}(h_{\Theta}(x), y) = -y \log(h_{\Theta}(x)) - (1-y) \log(1-h_{\Theta}(x))$$

If $y = 1$, $(1-y)$ term will become zero, therefore $-\log(h_{\Theta}(x))$ alone will be present

If $y = 0$, (y) term will become zero, therefore $-\log(1-h_{\Theta}(x))$ alone will be present

$$\begin{aligned}\text{Cost}(h_{\Theta}(x), Y(\text{Actual})) &= -\log(h_{\Theta}(x)) \text{ if } y=1 \\ &\quad -\log(1-h_{\Theta}(x)) \text{ if } y=0\end{aligned}$$

This implementation is for binary logistic regression. For data with more than 2 classes, softmax regression has to be used.

Q2. Difference between logistic and linear regression?

Answer:

Linear and Logistic regression are the most basic form of regression which are commonly used. The essential difference between these two is that Logistic regression is used when the dependent variable is binary. In contrast, Linear regression is used when the dependent variable is continuous, and the nature of the regression line is linear.

Key Differences between Linear and Logistic Regression

Linear regression models data using continuous numeric value. As against, logistic regression models the data in the binary values.

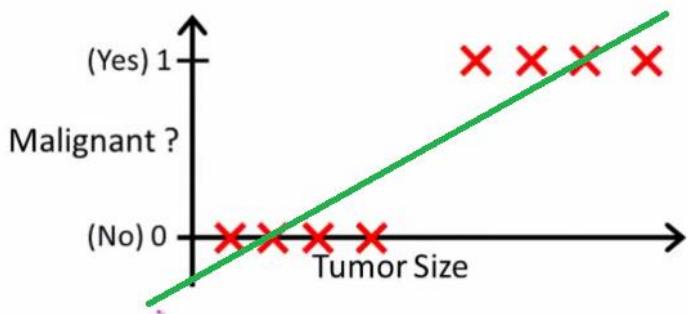
Linear regression requires to establish the linear relationship among dependent and independent variables, whereas it is not necessary for logistic regression.

In linear regression, the independent variable can be correlated with each other. On the contrary, in the logistic regression, the variable must not be correlated with each other.

Q3. Why we can't do a classification problem using Regression?

Answer:-

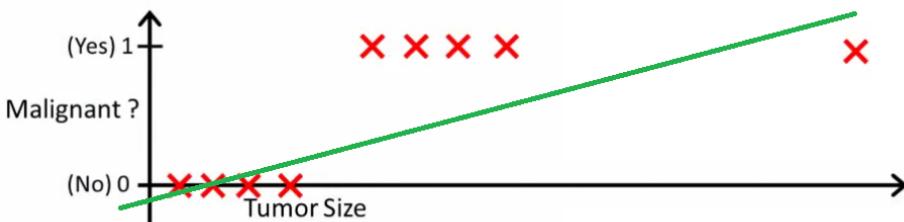
With linear regression you fit a polynomial through the data - say, like on the example below, we fit a straight line through {tumor size, tumor type} sample set:



Above, malignant tumors get 1, and non-malignant ones get 0, and the green line is our hypothesis $h(x)$. To make predictions, we may say that for any given tumor size x , if $h(x)$ gets bigger than 0.5, we predict malignant tumors. Otherwise, we predict benignly.

It looks like this way, we could correctly predict every single training set sample, but now let's change the task a bit.

Intuitively it's clear that all tumors larger certain threshold are malignant. So let's add another sample with huge tumor size, and run linear regression again:



Now our $h(x)>0.5 \rightarrow$ malignant doesn't work anymore. To keep making correct predictions, we need to change it to $h(x)>0.2$ or something - but that not how the algorithm should work.

We cannot change the hypothesis each time a new sample arrives. Instead, we should learn it off the training set data, and then (using the hypothesis we've learned) make correct predictions for the data we haven't seen before.

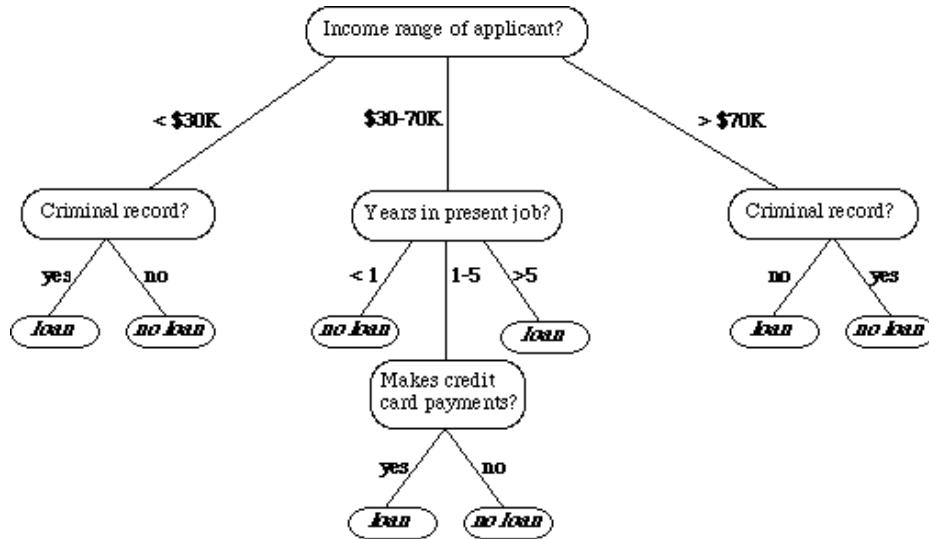
Linear regression is unbounded.

Q4. What is Decision Tree?

A decision tree is a type of supervised learning algorithm that can be used in classification as well as regressor problems. The input to a decision tree can be both continuous as well as categorical. The decision tree works on an if-then statement. Decision tree tries to solve a problem by using tree representation (Node and Leaf)

Assumptions while creating a decision tree: 1) Initially all the training set is considered as a root 2) Feature values are preferred to be categorical, if continuous then they are discretized 3) Records are

distributed recursively on the basis of attribute values 4) Which attributes are considered to be in root node or internal node is done by using a statistical approach.



Q5. Entropy, Information Gain, Gini Index, Reducing Impurity?

Answer:

There are different attributes which define the split of nodes in a decision tree. There are few algorithms to find the optimal split.

- 1) **ID3(Iterative Dichotomiser 3):** This solution uses Entropy and Information gain as metrics to form a better decision tree. The attribute with the highest information gain is used as a root node, and a similar approach is followed after that. Entropy is the measure that characterizes the impurity of an arbitrary collection of examples.

Entropy

Entropy $H(S)$ is a measure of the amount of uncertainty in the (data) set S (i.e. entropy characterizes the (data) set S).

$$H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$$

Where,

- S – The current (data) set for which entropy is being calculated (changes every iteration of the ID3 algorithm)
- C – Set of classes in S $C=\{\text{yes}, \text{no}\}$
- $p(c)$ – The proportion of the number of elements in class c to the number of elements in set S

When $H(S) = 0$, the set S is perfectly classified (i.e. all elements in S are of the same class).

In ID3, entropy is calculated for each remaining attribute. The attribute with the **smallest** entropy is used to split the set S on this iteration. The higher the entropy, the higher the potential to improve the classification here.

Entropy varies from 0 to 1. 0 if all the data belong to a single class and 1 if the class distribution is equal. In this way, entropy will give a measure of impurity in the dataset.

Steps to decide which attribute to split:

1. Compute the entropy for the dataset
2. For every attribute:
 - 2.1 Calculate entropy for all categorical values.
 - 2.2 Take average information entropy for the attribute.
 - 2.3 Calculate gain for the current attribute.
3. Pick the attribute with the highest information gain.
4. Repeat until we get the desired tree.

A leaf node is decided when entropy is zero

$$\text{Information Gain} = 1 - \sum (S_b/S) * \text{Entropy}(S_b)$$

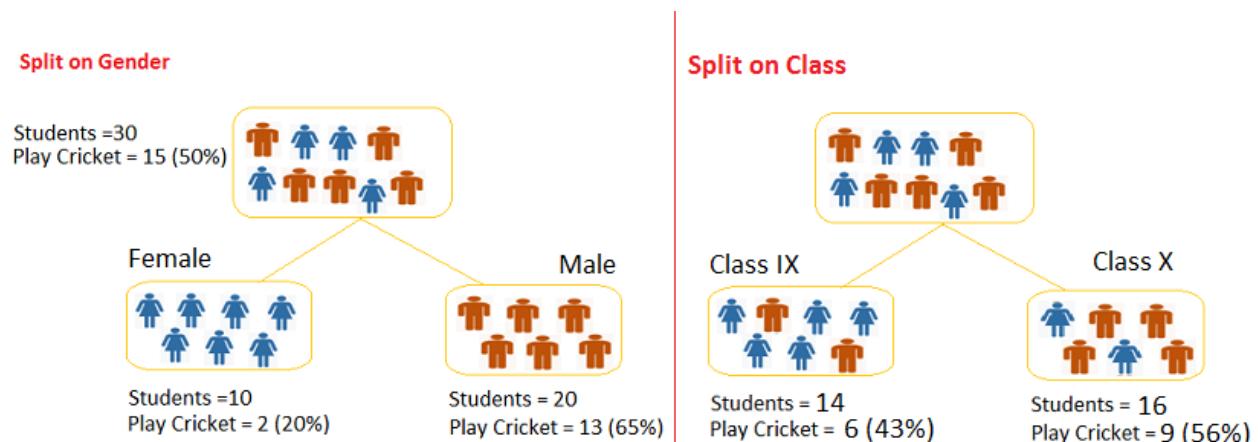
S_b - Subset, S - entire data

2) CART Algorithm (Classification and Regression trees): In CART, we use the GINI index as a metric. Gini index is used as a cost function to evaluate split in a dataset

Steps to calculate Gini for a split:

1. Calculate Gini for subnodes, using formula **sum of the square of probability for success and failure (p^2+q^2)**.
2. Calculate Gini for split using weighted Gini score of each node of that split.

Choose the split based on higher Gini value



Split on Gender:

$$\text{Gini for sub-node Female} = (0.2)*(0.2)+(0.8)*(0.8)=0.68$$

$$\text{Gini for sub-node Male} = (0.65)*(0.65)+(0.35)*(0.35)=0.55$$

Weighted Gini for Split Gender = $(10/30)*0.68+(20/30)*0.55 = 0.59$

Similar for Split on Class:

Gini for sub-node Class IX = $(0.43)*(0.43)+(0.57)*(0.57)=0.51$

Gini for sub-node Class X = $(0.56)*(0.56)+(0.44)*(0.44)=0.51$

Weighted Gini for Split Class = $(14/30)*0.51+(16/30)*0.51 = 0.51$

Here Weighted Gini is high for gender, so we consider splitting based on gender

Q6. How to control leaf height and Pruning?

Answer:

To control the leaf size, we can set the parameters:-

1. Maximum depth :

Maximum tree depth is a limit to stop the further splitting of nodes when the specified tree depth has been reached during the building of the initial decision tree.

NEVER use maximum depth to limit the further splitting of nodes. In other words: use the largest possible value.

2. Minimum split size:

Minimum split size is a limit to stop the further splitting of nodes when the number of observations in the node is lower than the minimum split size.

This is a good way to limit the growth of the tree. When a leaf contains too few observations, further splitting will result in overfitting (modeling of noise in the data).

3. Minimum leaf size

Minimum leaf size is a limit to split a node when the number of observations in one of the child nodes is lower than the minimum leaf size.

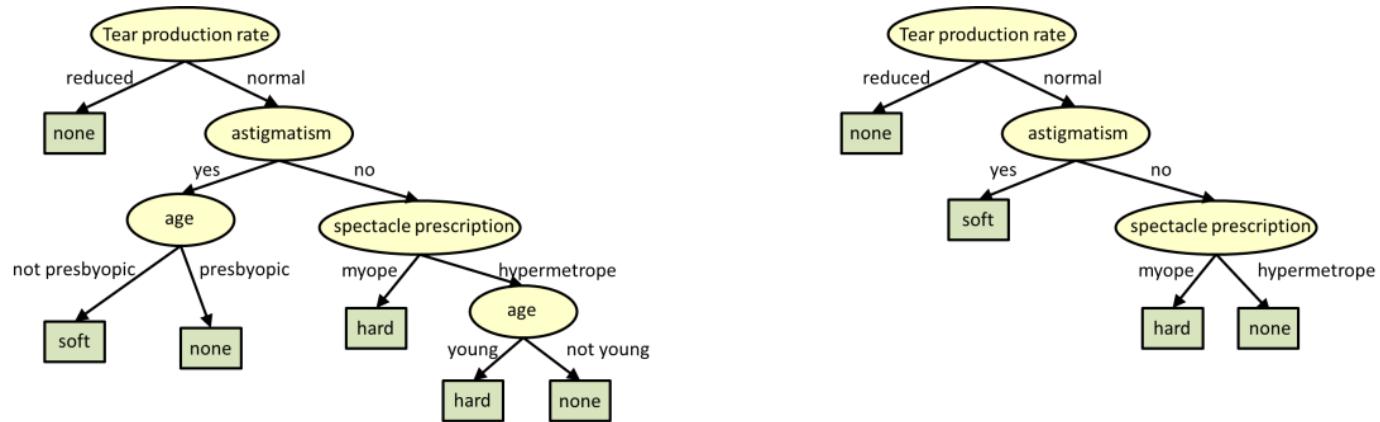
Pruning is mostly done to reduce the chances of overfitting the tree to the training data and reduce the overall complexity of the tree.

There are two types of pruning: **Pre-pruning** and **Post-pruning**.

1. Pre-pruning is also known as the **early stopping criteria**. As the name suggests, the criteria are set as parameter values while building the model. The tree stops growing when it meets any of these pre-pruning criteria, or it discovers the pure classes.

2. In Post-pruning, the idea is to allow the decision tree to grow fully and observe the CP value. Next, we prune/cut the tree with the optimal **CP**(Complexity Parameter) value as the parameter.

The CP (complexity parameter) is used to control tree growth. If the cost of adding a variable is higher, then the value of CP, tree growth stops.



Q7. How to handle a decision tree for numerical and categorical data?

Answer:

Decision trees can handle both categorical and numerical variables at the same time as features. There is not any problem in doing that.

Every split in a decision tree is based on a feature.

1. **If the feature is categorical, the split is done with the elements belonging to a particular class.**
2. **If the feature is continuous, the split is done with the elements higher than a threshold.**

At every split, the decision tree will take the best variable at that moment. This will be done according to an impurity measure with the split branches. And the fact that the variable used to do split is categorical or continuous is irrelevant (in fact, decision trees categorize continuous variables by creating binary regions with the threshold).

At last, the good approach is to always convert your **categoricals to continuous** using **LabelEncoder** or **OneHotEncoding**.

Q8. What is the Random Forest Algorithm?

Answer:

Random Forest is an ensemble machine learning algorithm that follows the bagging technique. The base estimators in the random forest are decision trees. Random forest randomly selects a set of features that are used to decide the best split at each node of the decision tree.

Looking at it step-by-step, this is what a random forest model does:

1. Random subsets are created from the original dataset (**bootstrapping**).
2. At each node in the decision tree, only a random set of features are considered to decide the best split.
3. A decision tree model is fitted on each of the subsets.
4. The final prediction is calculated by averaging the predictions from all decision trees.

To sum up, the Random forest randomly selects data points and features and builds multiple trees (Forest).

Random Forest is used for feature importance selection. The attribute (**.feature_importances_**) is used to find feature importance.

Some Important Parameters:-

1. **n_estimators**:- It defines the number of decision trees to be created in a random forest.
2. **criterion**:- "Gini" or "Entropy."
3. **min_samples_split**:- Used to define the minimum number of samples required in a leaf node before a split is attempted
4. **max_features**:- It defines the maximum number of features allowed for the split in each decision tree.
5. **n_jobs**:- The number of jobs to run in parallel for both fit and predict. **Always keep (-1) to use all the cores for parallel processing.**

Q9. What is Variance and Bias tradeoff?

Answer:

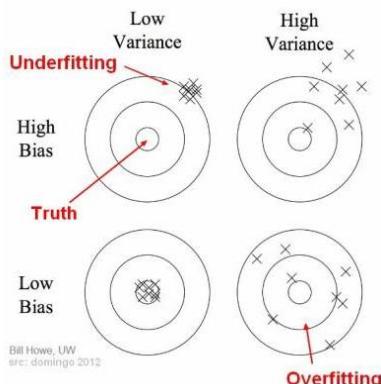
In predicting models, the prediction error is composed of two different errors

1. Bias
2. Variance

It is important to understand the variance and bias trade-off which tells about to minimize the Bias and Variance in the prediction and avoids overfitting & under fitting of the model.

Bias: It is the difference between the expected or average prediction of the model and the correct value which we are trying to predict. Imagine if we are trying to build more than one model by collecting different data sets, and later on, evaluating the prediction, we may end up by different prediction for all the models. So, bias is something which measures how far these model prediction from the correct prediction. It always leads to a high error in training and test data.

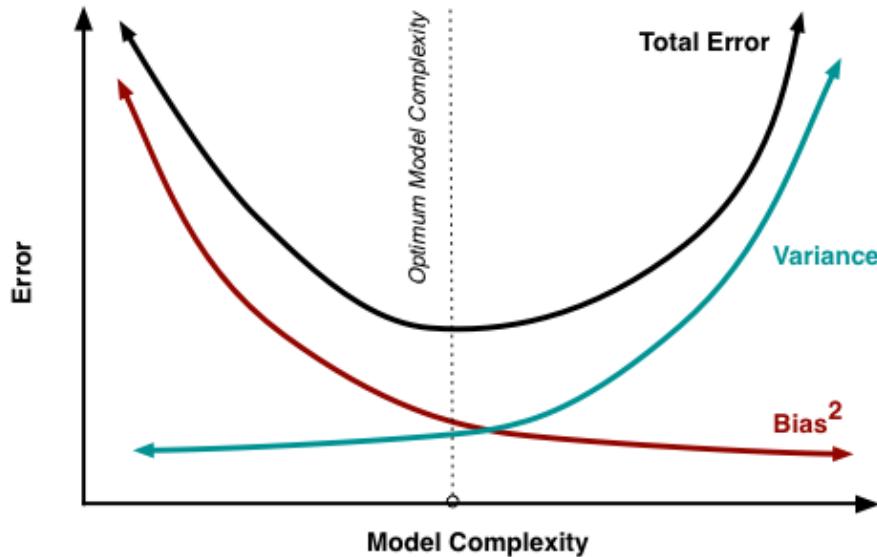
Variance: Variability of a model prediction for a given data point. We can build the model multiple times, so the variance is how much the predictions for a given point vary between different realizations of the model.



For example: Voting Republican - 13 Voting Democratic - 16 Non-Respondent - 21 Total - 50
 The probability of voting Republican is $13/(13+16)$, or 44.8%. We put out our press release that the Democrats are going to win by over 10 points; but, when the election comes around, it turns out they lose by 10 points. That certainly reflects poorly on us. Where did we go wrong in our model?

Bias scenario's: using a phonebook to select participants in our survey is one of our sources of bias. By only surveying certain classes of people, it skews the results in a way that will be consistent if we repeated the entire model building exercise. Similarly, not following up with respondents is another source of bias, as it consistently changes the mixture of responses we get. On our bulls-eye diagram, these move us away from the center of the target, but they would not result in an increased scatter of estimates.

Variance scenarios: the small sample size is a source of variance. If we increased our sample size, the results would be more consistent each time we repeated the survey and prediction. The results still might be highly inaccurate due to our large sources of bias, but the variance of predictions will be reduced



Q10. What are Ensemble Methods?

Answer

1. Bagging and Boosting

Decision trees have been around for a long time and also known to suffer from bias and variance. You will have a large bias with simple trees and a large variance with complex trees.

Ensemble methods - which combines several decision trees to produce better predictive performance than utilizing a single decision tree. The main principle behind the ensemble model is that a group of weak learners come together to form a strong learner.

Two techniques to perform ensemble decision trees:

1. Bagging
2. Boosting

Bagging (Bootstrap Aggregation) is used when our goal is to reduce the variance of a decision tree. Here the idea is to create several subsets of data from the training sample chosen randomly with replacement. Now, each collection of subset data is used to train their decision trees. As a result, we end up with an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree.

Boosting is another ensemble technique to create a collection of predictors. In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analyzing data

for errors. In other words, we fit consecutive trees (random sample), and at every step, the goal is to solve for net error from the prior tree.

When a hypothesis misclassifies an input, its weight is increased, so that the next hypothesis is more likely to classify it correctly. By combining the whole set at the end converts weak learners into a better performing model.

The different types of boosting algorithms are:

1. **AdaBoost**
2. **Gradient Boosting**
3. **XGBoost**

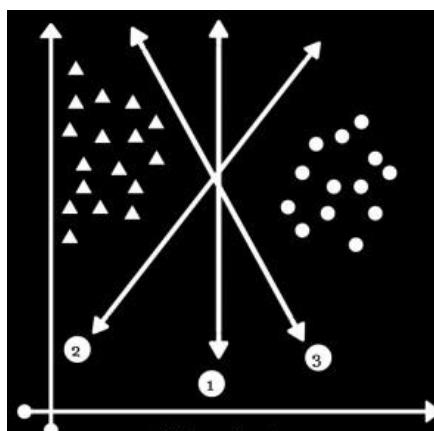
Q11. What is SVM Classification?

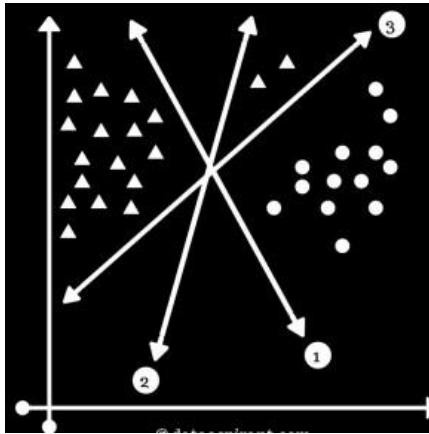
Answer:

SVM or Large margin classifier is a supervised learning algorithm that uses a powerful technique called SVM for classification.

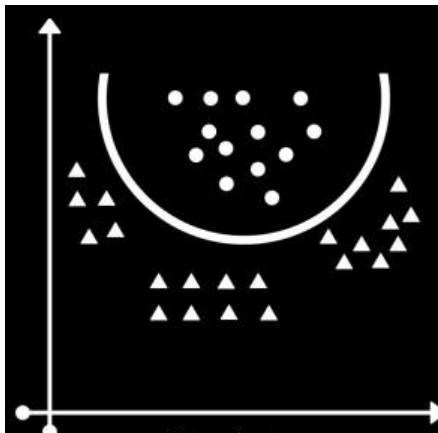
We have two types of SVM classifiers:

1) Linear SVM: In Linear SVM, the data points are expected to be separated by some apparent gap. Therefore, the SVM algorithm predicts a straight hyperplane dividing the two classes. The hyperplane is also called as maximum margin hyperplane





2) Non-Linear SVM: It is possible that our data points are not linearly separable in a p-dimensional space, but can be linearly separable in a higher dimension. Kernel tricks make it possible to draw nonlinear hyperplanes. Some standard kernels are a) Polynomial Kernel b) RBF kernel(mostly used).



Advantages of SVM classifier:

- 1) SVMs are effective when the number of features is quite large.
- 2) It works effectively even if the number of features is greater than the number of samples.
- 3) Non-Linear data can also be classified using customized hyperplanes built by using kernel trick.
- 4) It is a robust model to solve prediction problems since it maximizes margin.

Disadvantages of SVM classifier:

- 1) The biggest limitation of the Support Vector Machine is the choice of the kernel. The wrong choice of the kernel can lead to an increase in error percentage.
- 2) With a greater number of samples, it starts giving poor performances.
- 3) SVMs have good generalization performance, but they can be extremely slow in the test phase.
- 4) SVMs have high algorithmic complexity and extensive memory requirements due to the use of quadratic programming.

Q11. What is Naive Bayes Classification and Gaussian Naive Bayes

Answer:

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

↓ ↓
 THE PROBABILITY OF "B" BEING TRUE GIVEN THAT "A" IS TRUE THE PROBABILITY OF "A" BEING TRUE
 ↑ ↑
 THE PROBABILITY OF "A" BEING TRUE GIVEN THAT "B" IS TRUE THE PROBABILITY OF "B" BEING TRUE

Now, with regards to our dataset, we can apply Bayes' theorem in following way:

$$P(y|X) = \{P(X|y) P(y)\} / \{P(X)\}$$

where, y is class variable and X is a dependent feature vector (of size n) where:

$$X = (x_1, x_2, x_3, \dots, x_n)$$

	OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY GOLF
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

To clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of the dataset)

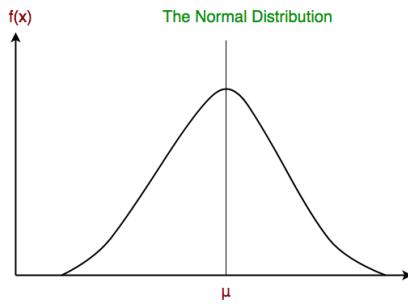
$X = (\text{Rainy}, \text{Hot}, \text{High}, \text{False})$ $y = \text{No}$ So basically, $P(X|y)$ here means, the probability of “Not playing golf” given that the weather conditions are “Rainy outlook”, “Temperature is hot”, “high humidity” and “no wind”.

Naive Bayes Classification:

1. We assume that no pair of features are dependent. For example, the temperature being ‘Hot’ has nothing to do with the humidity, or the outlook being ‘Rainy’ does not affect the winds. Hence, the features are assumed to be independent.
2. Secondly, each feature is given the same weight (or importance). For example, knowing the only temperature and humidity alone can’t predict the outcome accurately. None of the attributes is irrelevant and assumed to be contributing equally to the outcome

Gaussian Naive Bayes

Continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also called Normal distribution. When plotted, it gives a bell-shaped curve which is symmetric about the mean of the feature values as shown below:



This is as simple as calculating the mean and standard deviation values of each input variable (x) for each class value.

$$\text{Mean } (x) = 1/n * \text{sum}(x)$$

Where n is the number of instances, and x is the values for an input variable in your training data.

We can calculate the standard deviation using the following equation:

$$\text{Standard deviation}(x) = \sqrt{1/n * \text{sum}((x - \text{mean}(x))^2)}$$

When to use what? Standard Naive Bayes only supports categorical features, while Gaussian Naive Bayes only supports continuously valued features.

Q12. What is the Confusion Matrix?

Answer:

A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class.

This is the key to the confusion matrix.

It gives us insight not only into the errors being made by a classifier but, more importantly, the types of errors that are being made.

	<i>Class 1 Predicted</i>	<i>Class 2 Predicted</i>
<i>Class 1 Actual</i>	TP	FN
<i>Class 2 Actual</i>	FP	TN

Here,

- Class 1: Positive
- Class 2: Negative

Definition of the Terms:

1. **Positive (P):** Observation is positive (for example: is an apple).
2. **Negative (N):** Observation is not positive (for example: is not an apple).
3. **True Positive (TP):** Observation is positive, and is predicted to be positive.
4. **False Negative (FN):** Observation is positive, but is predicted negative.
5. **True Negative (TN):** Observation is negative, and is predicted to be negative.
6. **False Positive (FP):** Observation is negative, but is predicted positive.

Q13. What is Accuracy and Misclassification Rate?

Answer:

Accuracy

Accuracy is defined as the ratio of the sum of True Positive and True Negative by Total(TP+TN+FP+FN)

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

However, there are problems with accuracy. It assumes equal costs for both kinds of errors. A 99% accuracy can be excellent, good, mediocre, poor, or terrible depending upon the problem.

Misclassification Rate

Misclassification Rate is defined as the ratio of the sum of False Positive and False Negative by Total(TP+TN+FP+FN)

Misclassification Rate is also called Error Rate.

	Actual Positive	Actual Negative
Predicted Positive	True Positive(TP)	False Positive(FP) (Type 1 Error)
Predicted Negative	False Negative(FN) (Type 2 Error)	True Negative(TN)

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total Population}}$$

$$\text{Error Rate/Misclassification rate} = \frac{\text{False Positive} + \text{False Negative}}{\text{Total Population}}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{Predicted Positive}(TP+FP)}$$

$$\text{Sensitivity/Recall} = \frac{\text{True Positive}}{\text{Actual Positive}(TP+FN)}$$

$$\text{Specificity} = \frac{\text{True Negative}}{\text{Actual Negative}(FP+TN)}$$

$$\text{F1 Score} = \frac{2 * (\text{Recall} * \text{Precision})}{\text{Recall} + \text{Precision}}$$

Q14. True Positive Rate & True Negative Rate

Answer:

True Positive Rate:

Sensitivity (SN) is calculated as the number of correct positive predictions divided by the total number of positives. It is also called **Recall (REC)** or true positive rate (TPR). The best sensitivity is 1.0, whereas the worst is 0.0.

$$SN = \frac{TP}{TP+FN} = \frac{TP}{P}$$

True Negative Rate

Specificity (SP) is calculated as the number of correct negative predictions divided by the total number of negatives. It is also called a true negative rate (TNR). The best specificity is 1.0, whereas the worst is 0.0.

$$SN = \frac{TP}{TPFN} = \frac{TP}{P}$$

Q15. What is False Positive Rate & False negative Rate?

False Positive Rate

False positive rate (FPR) is calculated as the number of incorrect positive predictions divided by the total number of negatives. The best false positive rate is 0.0, whereas the worst is 1.0. It can also be calculated as $1 - \text{specificity}$.

$$SN = \frac{TP}{TPFN} = \frac{TP}{P}$$

False Negative Rate

False Negative rate (FPR) is calculated as the number of incorrect positive predictions divided by the total number of positives. The best false negative rate is 0.0, whereas the worst is 1.0.

Name	Formula	Explanation
True Positive Rate (TP rate)	$TP / (TP + FP)$	The closer to 1, the better. TP rate = 1 when FP = 0. (No false positives)
True Negative Rate (TN rate)	$TN / (TN + FN)$	The closer to 1, the better. TN rate = 1 when FN = 0. (No false negatives)
False Positive Rate (FP rate)	$FP / (FP + TN)$	The closer to 0, the better. FP rate = 0 when FP = 0. (No false positives)
False Negative Rate (FN rate)	$FN / (FN + TP)$	The closer to 0, the better. FN rate = 0 when FN = 0. (No false negatives)

Q16. What are F1 Score, precision and recall?

Recall:-

Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples.

1. High Recall indicates the class is correctly recognized (small number of FN).
2. Low Recall indicates the class is incorrectly recognized (large number of FN).

Recall is given by the relation:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Precision:

To get the value of precision, we divide the total number of correctly classified positive examples by the total number of predicted positive examples.

1. High Precision indicates an example labeled as positive is indeed positive (a small number of FP).
2. Low Precision indicates an example labeled as positive is indeed positive (large number of FP).

The relation gives precision:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Remember:-

High recall, low precision: This means that most of the positive examples are correctly recognized (low FN), but there are a lot of false positives.

Low recall, high precision: This shows that we miss a lot of positive examples (high FN), but those we predict as positive are indeed positive (low FP).

F-measure/F1-Score:

Since we have two measures (Precision and Recall), it helps to have a measurement that represents both of them. We calculate an **F-measure**, which uses **Harmonic Mean in place of Arithmetic Mean as it punishes the extreme values more.**

The F-Measure will always be nearer to the smaller value of Precision or Recall.

$$F\text{-measure} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

Q17. What is RandomizedSearchCV?

Answer:

Randomized search CV is used to perform a random search on hyperparameters. Randomized search CV uses a fit and score method, predict_proba, decision_func, transform, etc.,

The parameters of the estimator used to apply these methods are optimized by cross-validated search over parameter settings.

In contrast to GridSearchCV, not all parameter values are tried out, but rather a fixed number of parameter settings is sampled from the specified distributions. The number of parameter settings that are tried is given by n_iter.

Code Example :

```
class sklearn.model_selection.RandomizedSearchCV(estimator, param_distributions,
n_iter=10, scoring=None, fit_params=None, n_jobs=None, iid='warn', refit=True,
cv='warn', verbose=0, pre_dispatch='2n_jobs', random_state=None, error_score='raise-
deprecating', return_train_score='warn')
```

Q18. What is GridSearchCV?

Answer:

Grid search is the process of performing hyperparameter tuning to determine the optimal values for a given model.

CODE Example:-

```
from sklearn.model_selection import GridSearchCV from sklearn.svm import SVR gsc = GridSearchCV( estimator=SVR(kernel='rbf'), param_grid={ 'C': [0.1, 1, 100, 1000], 'epsilon': [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10], 'gamma': [0.0001, 0.001, 0.005, 0.1, 1, 3, 5] }, cv=5, scoring='neg_mean_squared_error', verbose=0, n_jobs=-1)
```

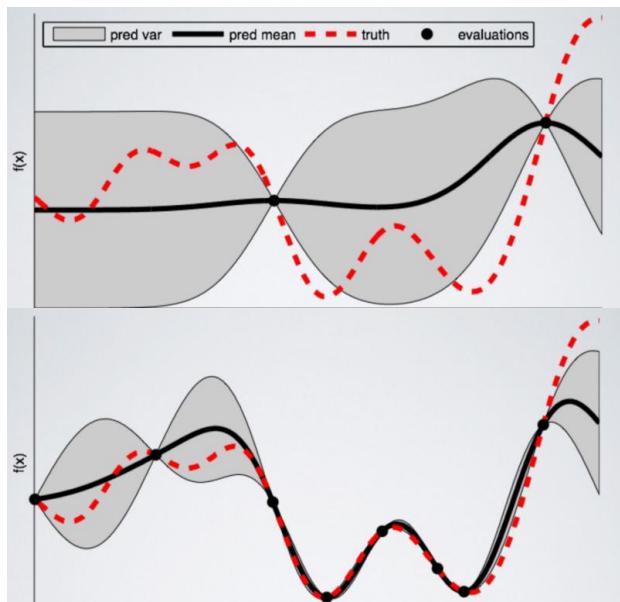
Grid search runs the model on all the possible range of hyperparameter values and outputs the best model

Q19. What is BayesianSearchCV?

Answer:

Bayesian search, in contrast to the grid and random search, keeps track of past evaluation results, which they use to form a probabilistic model mapping hyperparameters to a probability of a score on the objective function.

$$P(score | \text{hyperparameters})$$



Code:

```
from skopt import BayesSearchCV
opt = BayesSearchCV(
    SVC(),
```

```
{  
    'C': (1e-6, 1e+6, 'log-uniform'),  
    'gamma': (1e-6, 1e+1, 'log-uniform'),  
    'degree': (1, 8), # integer valued parameter  
    'kernel': ['linear', 'poly', 'rbf']  
},  
n_iter=32,  
cv=3)
```

Q20. What is ZCA Whitening?

Answer:

Zero Component Analysis:

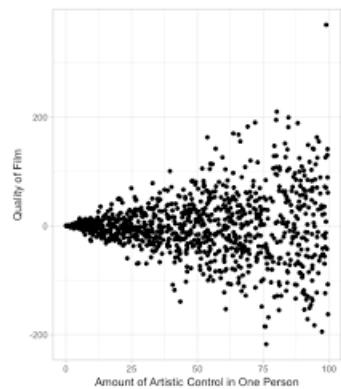
Making the co-variance matrix as the Identity matrix is called whitening. This will remove the first and second-order statistical structure

ZCA transforms the data to zero means and makes the features linearly independent of each other

In some image analysis applications, especially when working with images of the color and tiny type, it is frequently interesting to apply some whitening to the data before, e.g. training a classifier.

Q1. How do you treat heteroscedasticity in regression?

Heteroscedasticity means unequal scattered distribution. In regression analysis, we generally talk about the heteroscedasticity in the context of the error term. Heteroscedasticity is the systematic change in the spread of the residuals or errors over the range of measured values. Heteroscedasticity is the problem because *Ordinary least squares (OLS)* regression assumes that all residuals are drawn from a random population that has a constant variance.



What causes Heteroscedasticity?

Heteroscedasticity occurs more often in datasets, where we have a large range between the largest and the smallest observed values. There are many reasons why heteroscedasticity can exist, and a generic explanation is that the error variance changes proportionally with a factor.

We can categorize Heteroscedasticity into two general types:-

Pure heteroscedasticity:- It refers to cases where we specify the correct model and let us observe the non-constant variance in residual plots.

Impure heteroscedasticity:- It refers to cases where you incorrectly specify the model, and that causes the non-constant variance. When you leave an important variable out of a model, the omitted effect is absorbed into the error term. If the effect of the omitted variable varies throughout the observed range of data, it can produce the telltale signs of heteroscedasticity in the residual plots.

How to Fix Heteroscedasticity

Redefining the variables:

If your model is a cross-sectional model that includes large differences between the sizes of the observations, you can find different ways to specify the model that reduces the impact of the size

differential. To do this, change the model from using the raw measure to using rates and per capita values. Of course, this type of model answers a slightly different kind of question. You'll need to determine whether this approach is suitable for both your data and what you need to learn.

Weighted regression:

It is a method that assigns each data point to a weight based on the variance of its fitted value. The idea is to give small weights to observations associated with higher variances to shrink their squared residuals. Weighted regression minimizes the sum of the weighted squared residuals. When you use the correct weights, heteroscedasticity is replaced by homoscedasticity.

Q2. What is multicollinearity, and how do you treat it?

Multicollinearity means independent variables are highly correlated to each other. In regression analysis, it's an important assumption that the regression model should not be faced with a problem of multicollinearity.

If two explanatory variables are highly correlated, it's hard to tell, which affects the dependent variable. Let's say Y is regressed against X1 and X2 and where X1 and X2 are highly correlated. Then the effect of X1 on Y is hard to distinguish from the effect of X2 on Y because any increase in X1 tends to be associated with an increase in X2.

Another way to look at the multicollinearity problem is: Individual t-test P values can be misleading. It means a P-value can be high, which means the variable is not important, even though the variable is important.

Correcting Multicollinearity:

- 1) Remove one of the highly correlated independent variables from the model. If you have two or more factors with a high VIF, remove one from the model.
- 2) Principle Component Analysis (PCA) - It cut the number of interdependent variables to a smaller set of uncorrelated components. Instead of using highly correlated variables, use components in the model that have eigenvalue greater than 1.
- 3) Run PROC VARCLUS and choose the variable that has a minimum (1-R²) ratio within a cluster.
- 4) Ridge Regression - It is a technique for analyzing multiple regression data that suffer from multicollinearity.
- 5) If you include an interaction term (the product of two independent variables), you can also reduce multicollinearity by "centering" the variables. By "centering," it means subtracting the mean from the values of the independent variable before creating the products.

When is multicollinearity not a problem?

- 1) If your goal is to predict Y from a set of X variables, then multicollinearity is not a problem. The predictions will still be accurate, and the overall R² (or adjusted R²) quantifies how well the model predicts the Y values.
- 2) Multiple dummy (binary) variables that represent a categorical variable with three or more categories.

Q3. What is market basket analysis? How would you do it in Python?

Market basket analysis is the study of items that are purchased or grouped in a single transaction or multiple, sequential transactions. Understanding the relationships and the strength of those relationships is valuable information that can be used to make recommendations, cross-sell, up-sell, offer coupons, etc.

Market Basket Analysis is one of the key techniques used by large retailers to uncover associations between items. It works by looking for combinations of items that occur together frequently in transactions. To put it another way, it allows retailers to identify relationships between the items that people buy.

Q4. What is Association Analysis? Where is it used?

Association analysis uses a set of transactions to discover rules that indicate the likely occurrence of an item based on the occurrences of other items in the transaction.

The technique of association rules is widely used for retail basket analysis. It can also be used for classification by using rules with class labels on the right-hand side. It is even used for outlier detection with rules indicating infrequent/abnormal association.

Association analysis also helps us to identify cross-selling opportunities, for example, we can use the rules resulting from the analysis to place associated products together in a catalog, in the supermarket, or the Webshop, or apply them when targeting a marketing campaign for product B at customers who have already purchased product A.

Association rules are given in the form as below:

A=>B[Support,Confidence] The part before => is referred to as if (Antecedent) and the part after => is referred to as then (Consequent).

Where A and B are sets of items in the transaction data, a and B are disjoint sets.

Computer=>Anti-virusSoftware[Support=20%,confidence=60%] Above rule says:

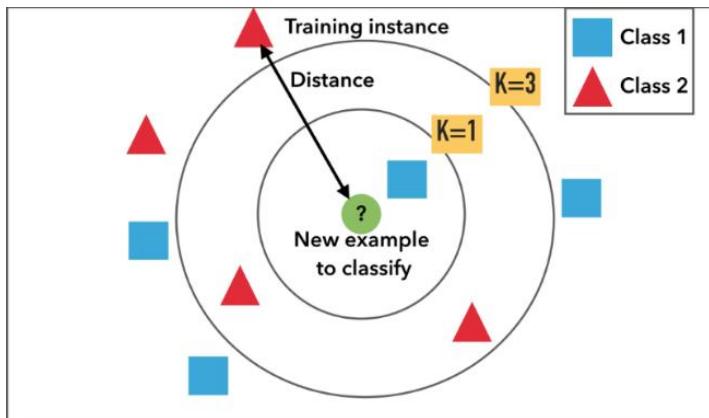
1. 20% transaction show Anti-virus software is bought with purchase of a Computer
2. 60% of customers who purchase Anti-virus software is bought with purchase of a Computer

An example of Association Rules * Assume there are 100 customers

1. 10 of them bought milk, 8 bought butter and 6 bought both of them 2 .bought milk => bought butter
2. support = $P(\text{Milk} \& \text{Butter}) = 6/100 = 0.06$
3. confidence = support/ $P(\text{Butter}) = 0.06/0.08 = 0.75$
4. lift = confidence/ $P(\text{Milk}) = 0.75/0.10 = 7.5$

Q5. What is KNN Classifier ?

KNN means **K-Nearest Neighbour** Algorithm. It can be used for both classification and regression.



It is the simplest machine learning algorithm. Also known as **lazy learning** (why? Because it does not create a generalized model during the time of training, so the testing phase is very important where it does the actual job. Hence Testing is very costly - in terms of time & money). Also called an instance-based or memory-based learning

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is assigned to the class of that single nearest neighbor.

In **k-NN regression**, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

Distance functions

Euclidean

$$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

Manhattan

$$\sum_{i=1}^k |x_i - y_i|$$

Minkowski

$$\left(\sum_{i=1}^k (|x_i - y_i|)^q \right)^{1/q}$$

All three distance measures are only valid for continuous variables. In the instance of categorical variables, the Hamming distance must be used.

Hamming Distance

$$D_H = \sum_{i=1}^k |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

How to choose the value of K: K value is a hyperparameter which needs to choose during the time of model building

Also, a small number of neighbors are most flexible fit, which will have a low bias, but the high variance and a large number of neighbors will have a smoother decision boundary, which means lower variance but higher bias.

We should choose an odd number if the number of classes is even. It is said the most common values are to be 3 & 5.

Q6. What is Pipeline in sklearn ?

A pipeline is what chains several steps together, once the initial exploration is done. For example, some codes are meant to transform features—normalize numerically, or turn text into vectors, or fill up missing data, and they are transformers; other codes are meant to predict variables by fitting an algorithm,

such as random forest or support vector machine, they are estimators. Pipeline chains all these together, which can then be applied to training data in block.

Example of a pipeline that imputes data with the most frequent value of each column, and then fit a decision tree classifier.

```
From sklearn.pipeline import Pipeline  
steps = [('imputation', Imputer(missing_values='NaN', strategy = 'most_frequent', axis=0)),  
         ('clf', DecisionTreeClassifier())]  
pipeline = Pipeline(steps)  
clf = pipeline.fit(X_train,y_train)``
```

Instead of fitting to one model, it can be looped over several models to find the best one.

```
classifiers = [ KNeighborsClassifier(5), RandomForestClassifier(), GradientBoostingClassifier()]  
for clf in classifiers:  
    steps = [('imputation', Imputer(missing_values='NaN', strategy = 'most_frequent', axis=0)),  
             ('clf', clf)]  
  
    pipeline = Pipeline(steps)
```

I also learned the pipeline itself can be used as an estimator and passed to cross-validation or grid search.

```
from sklearn.model_selection import KFold  
from sklearn.model_selection import cross_val_score  
kfolds = KFold(n_splits=10, random_state=seed)  
results = cross_val_score(pipeline, X_train, y_train, cv=kfolds)  
print(results.mean())
```

Q7. What is Principal Component Analysis(PCA), and why we do?

The main idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. The same is done by transforming the variables to a new set of variables, which are known as the principal components (or simply, the PCs) and are orthogonal, ordered such that the retention of variation present in the original variables decreases as we move down in the order. So, in this way, the 1st principal component retains maximum variation that was present in the original components. The principal components are the eigenvectors of a covariance matrix, and hence they are orthogonal.

Main important points to be considered:

1. Normalize the data
2. Calculate the covariance matrix
3. Calculate the eigenvalues and eigenvectors
4. Choosing components and forming a feature vector
5. Forming Principal Components

Q8. What is t-SNE?

(t-SNE) t-Distributed Stochastic Neighbor Embedding is a non-linear dimensionality reduction algorithm used for exploring high-dimensional data. It maps multi-dimensional data to two or more dimensions suitable for human observation. With the help of the t-SNE algorithms, you may have to plot fewer exploratory data analysis plots next time you work with high dimensional data.

Q9. VIF(Variation Inflation Factor),Weight of Evidence & Information

Value. Why and when to use?

Variation Inflation Factor

It provides an index that measures how much the variance (the square of the estimate's standard deviation) of an estimated regression coefficient is increased because of collinearity.

$VIF = 1 / (1 - R^2 \text{ of } j\text{-th variable})$ where R^2 of j th variable is the coefficient of determination of the model that includes all independent variables except the j th predictor.

Where R^2 of j -th variable is the multiple R^2 for the regression of X_j on the other independent variables (a regression that does not involve the dependent variable Y).

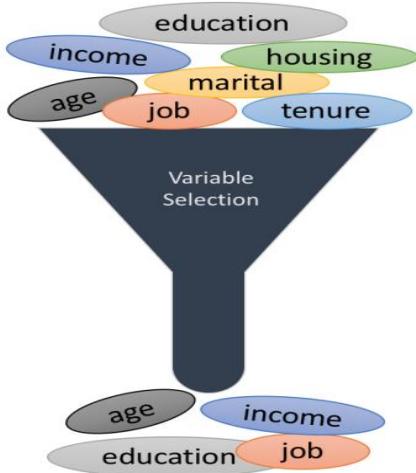
If $VIF > 5$, then there is a problem with multicollinearity.

Understanding VIF

If the variance inflation factor of a predictor variable is 5 this means that variance for the coefficient of that predictor variable is 5 times as large as it would be if that predictor variable were uncorrelated with the other predictor variables.

In other words, if the variance inflation factor of a predictor variable is 5 this means that the standard error for the coefficient of that predictor variable is $\sqrt{5} = 2.23$ times as large as it would be if that predictor variable were uncorrelated with the other predictor variables.

Weight of evidence (WOE) and information value (IV) are simple, yet powerful techniques to perform variable transformation and selection.



The formula to create WOE and IV is

$$WOE = \ln\left(\frac{\text{Event}\%}{\text{Non Event}\%}\right)$$

$$IV = \sum (\text{Event}\% - \text{Non Event}\%) * \ln\left(\frac{\text{Event}\%}{\text{Non Event}\%}\right)$$

Here is a simple table that shows how to calculate these values.

Variable Name	Min. Value	Max. Value	Count	# Event	# Non Event	Event%	Non event%	WOE	Event% - Non event%	IV
Age	10	20	1200	150	1050	28.3%	19.0%	0.3992	9.3%	0.03718
Age	21	30	900	120	780	22.6%	14.1%	0.4733	8.5%	0.04040
Age	31	40	1090	110	980	20.8%	17.7%	0.1580	3.0%	0.00479
Age	41	50	1460	100	1360	18.9%	24.6%	-0.2650	-5.7%	0.01517
Age	50	inf	1410	50	1360	9.4%	24.6%	-0.9582	-15.2%	0.14525
Total			6060	530	5530					0.24279

The IV value can be used to select variables quickly.

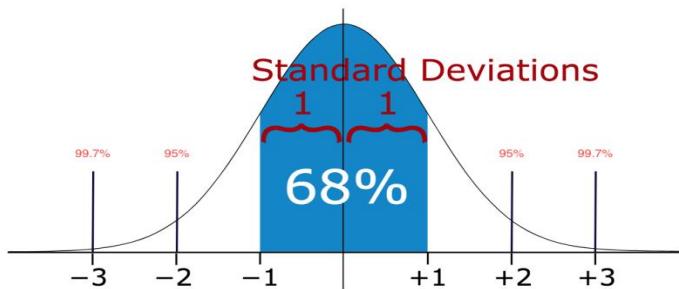
Information Value (IV)	Predictive Power
< 0.02	useless for prediction
0.02 to 0.1	weak predictor
0.1 to 0.3	medium predictor
0.3 to 0.5	strong predictor
> 0.5	suspicious or too good to be true

Q10: How to evaluate that data does not have any outliers ?

In statistics, outliers are data points that don't belong to a certain population. It is an abnormal observation that lies far away from other values. An outlier is an observation that diverges from otherwise well-structured data.

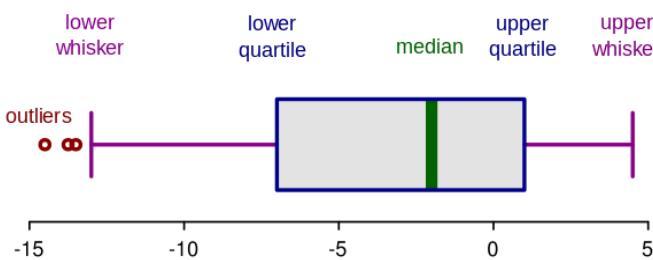
Detection:

Method 1 — Standard Deviation: In statistics, If a data distribution is approximately normal, then about 68% of the data values lie within one standard deviation of the mean, and about 95% are within two standard deviations, and about 99.7% lie within three standard deviations.

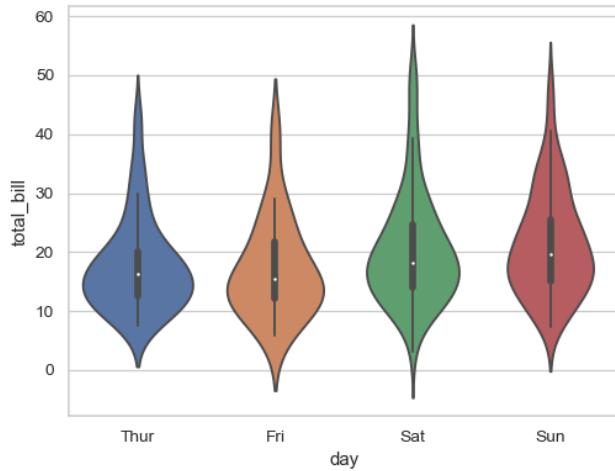


Therefore, if you have any data point that is more than 3 times the standard deviation, then those points are very likely to be anomalous or outliers.

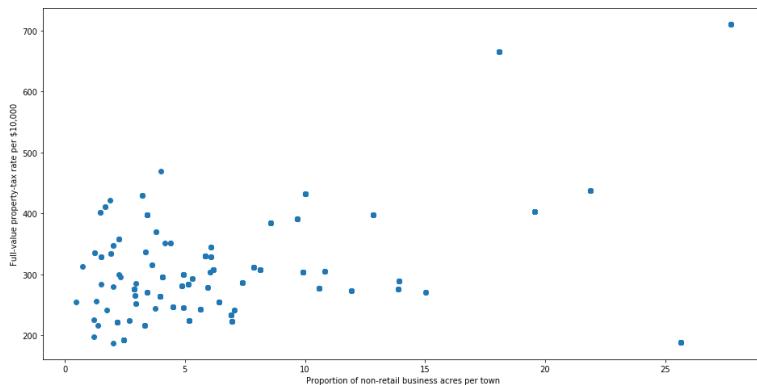
Method 2 — Boxplots: Box plots are a graphical depiction of numerical data through their quantiles. It is a very simple but effective way to visualize outliers. Think about the lower and upper whiskers as the boundaries of the data distribution. Any data points that show above or below the whiskers can be considered outliers or anomalous.



Method 3 - Violin Plots: Violin plots are similar to box plots, except that they also show the probability density of the data at different values, usually smoothed by a kernel density estimator. Typically a violin plot will include all the data that is in a box plot: a marker for the median of the data, a box or marker indicating the interquartile range, and possibly all sample points if the number of samples is not too high.



Method 4 - Scatter Plots: A scatter plot is a type of plot or mathematical diagram using Cartesian coordinates to display values for typically two variables for a set of data. The data are displayed as a collection of points, each having the value of one variable determining the position on the horizontal axis and the value of the other variable determining the position on the vertical axis.



The points which are very far away from the general spread of data and have a very few neighbors are considered to be outliers

Q11: What you do if there are outliers?

Following are the approaches to handle the outliers:

1. Drop the outlier records
2. Assign a new value: If an outlier seems to be due to a mistake in your data, you try imputing a value.
3. If percentage-wise the number of outliers is less, but when we see numbers, there are several, then, in that case, dropping them might cause a loss in insight. We should group them in that case and run our analysis separately on them.

Q12: What are the encoding techniques you have applied with Examples ?

In many practical data science activities, the data set will contain categorical variables. These variables are typically stored as text values". Since machine learning is based on mathematical equations, it would cause a problem when we keep categorical variables as is.

Let's consider the following dataset of fruit names and their weights.

Some of the common encoding techniques are:

Label encoding: In label encoding, we map each category to a number or a label. The labels chosen for the categories have no relationship. So categories that have some ties or are close to each other lose such information after encoding.

One - hot encoding: In this method, we map each category to a vector that contains 1 and 0 denoting the presence of the feature or not. The number of vectors depends on the categories which we want to keep. For high cardinality features, this method produces a lot of columns that slows down the learning significantly.

Q13: Tradeoff between bias and variances, the relationship between them.

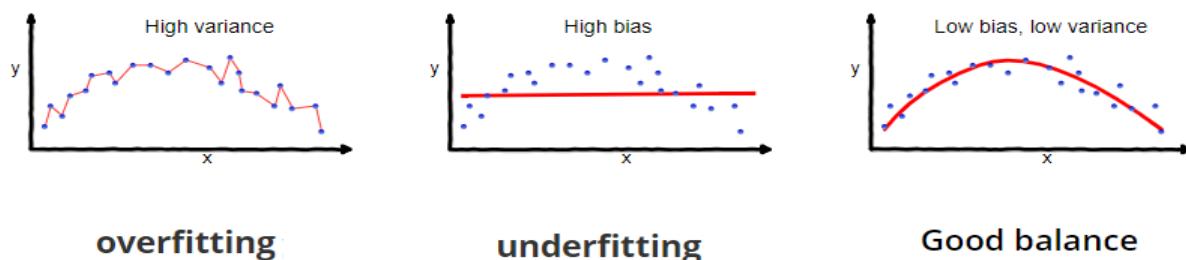
Whenever we discuss model prediction, it's important to understand prediction errors (bias and variance). The prediction error for any machine learning algorithm can be broken down into three parts:

- Bias Error
- Variance Error
- Irreducible Error

The irreducible error cannot be reduced regardless of what algorithm is used. It is the error introduced from the chosen framing of the problem and may be caused by factors like unknown variables that influence the mapping of the input variables to the output variable.

Bias: Bias means that the model favors one result more than the others. Bias is the simplifying assumptions made by a model to make the target function easier to learn. The model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to a high error in training and test data.

Variance: Variance is the amount that the estimate of the target function will change if different training data was used. The model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but have high error rates on test data.



So, the end goal is to come up with a model that balances both Bias and Variance. This is called *Bias Variance Trade-off*. To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.

Q14: What is the difference between Type 1 and Type 2 error and severity of the error?

Type I Error

A Type I error is often referred to as a "false positive" and is the incorrect rejection of the true null hypothesis in favor of the alternative.

In the example above, the null hypothesis refers to the natural state of things or the absence of the tested effect or phenomenon, i.e., stating that the patient is HIV negative. The alternative hypothesis states that the patient is HIV positive. Many medical tests will have the disease they are testing for as the alternative hypothesis and the lack of that disease as the null hypothesis.

A Type I error would thus occur when the patient doesn't have the virus, but the test shows that they do. In other words, the test incorrectly rejects the true null hypothesis that the patient is HIV negative.

Type II Error

A Type II error is the inverse of a Type I error and is the false acceptance of a null hypothesis that is not true, i.e., a false negative. A Type II error would entail the test telling the patient they are free of HIV when they are not.

Considering this HIV example, which error type do you think is more acceptable? In other words, would you rather have a test that was more prone to Type I or Types II error? With HIV, the momentary stress of a false positive is likely better than feeling relieved at a false negative and then failing to take steps to treat the disease. Pregnancy tests, blood tests, and any diagnostic tool that has serious consequences for the health of a patient are usually overly sensitive for this reason – they should err on the side of a false positive.

But in most fields of science, Type II errors are seen as less serious than Type I errors. With the Type II error, a chance to reject the null hypothesis was lost, and no conclusion is inferred from a non-rejected null. But the Type I error is more serious because you have wrongly rejected the null hypothesis and ultimately made a claim that is not true. In science, finding a phenomenon where there is none is more egregious than failing to find a phenomenon where there is.

Q15: What is binomial distribution and polynomial distribution?

Binomial Distribution: A binomial distribution can be thought of as simply the probability of a SUCCESS or FAILURE outcome in an experiment or survey that is repeated multiple times. The binomial is a type of distribution that has two possible outcomes (the prefix “bi” means two, or twice). For example, a coin toss has only two possible outcomes: heads or tails, and taking a test could have two possible outcomes: pass or fail.

Multimonial/Polynomial Distribution: Multi or Poly means many. In probability theory, the multinomial distribution is a generalization of the binomial distribution. For example, it models the probability of counts of each side for rolling a k-sided die n times. For n independent trials each of which leads to success for exactly one of k categories, with each category having a given fixed success probability, the multinomial distribution gives the probability of any particular combination of numbers of successes for the various categories

Q16: What is the Mean Median Mode standard deviation for the sample and population?

Mean It is an important technique in statistics. Arithmetic Mean can also be called an average. It is the number of the quantity obtained by summing two or more numbers/variables and then dividing the sum by the number of numbers/variables.

Mode The mode is also one of the types for finding the average. A mode is a number that occurs most frequently in a group of numbers. Some series might not have any mode; some might have two modes, which is called a bimodal series.

In the study of statistics, the three most common ‘averages’ in statistics are mean, median, and mode.

Median is also a way of finding the average of a group of data points. It’s the middle number of a set of numbers. There are two possibilities, the data points can be an odd number group, or it can be an even number group.

If the group is odd, arrange the numbers in the group from smallest to largest. The median will be the one which is exactly sitting in the middle, with an equal number on either side of it. If the group is even, arrange the numbers in order and pick the two middle numbers and add them then divide by 2. It will be the median number of that set.

Standard Deviation (Sigma) Standard Deviation is a measure of how much your data is spread out in statistics.

Q17: What is Mean Absolute Error ?

What is Absolute Error? Absolute Error is the amount of error in your measurements. It is the difference between the measured value and the “true” value. For example, if a scale states 90 pounds, but you know your true weight is 89 pounds, then the scale has an absolute error of $90\text{ lbs} - 89\text{ lbs} = 1\text{ lbs}$.

This can be caused by your scale, not measuring the exact amount you are trying to measure. For example, your scale may be accurate to the nearest pound. If you weigh 89.6 lbs, the scale may “round up” and give you 90 lbs. In this case the absolute error is $90\text{ lbs} - 89.6\text{ lbs} = .4\text{ lbs}$.

Mean Absolute Error The Mean Absolute Error(MAE) is the average of all absolute errors. The formula is: mean absolute error

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |x_i - \bar{x}|$$

Where,

n = the number of errors, Σ = summation symbol (which means “add them all up”), $|x_i - \bar{x}|$ = the absolute errors. The formula may look a little daunting, but the steps are easy:

Find all of your absolute errors, $x_i - \bar{x}$. Add them all up. Divide by the number of errors. For example, if you had 10 measurements, divide by 10.

Q18: What is the difference between long data and wide data?

There are many different ways that you can present the same dataset to the world. Let's take a look at one of the most important and fundamental distinctions, whether a dataset is wide or long.

The difference between wide and long datasets boils down to whether we prefer to have more columns in our dataset or more rows.

Wide Data A dataset that emphasizes putting additional data about a single subject in columns is called a wide dataset because, as we add more columns, the dataset becomes wider.

Long Data Similarly, a dataset that emphasizes including additional data about a subject in rows is called a long dataset because, as we add more rows, the dataset becomes longer. It's important to point out that there's nothing inherently good or bad about wide or long data.

In the world of data wrangling, we sometimes need to make a long dataset wider, and we sometimes need to make a wide dataset longer. However, it is true that, as a general rule, data scientists who embrace the concept of tidy data usually prefer longer datasets over wider ones.

Q19: What are the data normalization method you have applied, and why?

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges.

In simple words, when multiple attributes are there, but attributes have values on different scales, this may lead to poor data models while performing data mining operations. So they are normalized to bring all the attributes on the same scale, usually something between (0,1).

It is not always a good idea to normalize the data since we might lose information about maximum and minimum values. Sometimes it is a good idea to do so.

For example, ML algorithms such as Linear Regression or Support Vector Machines typically converge faster on normalized data. But on algorithms like K-means or K Nearest Neighbours, normalization could

be a good choice or a bad depending on the use case since the distance between the points plays a key role here.

person_name	Salary	Year_of_experience	Expected Position Level	
Aman	100000	10	2	
Abhinav	78000	7	4	
Ashutosh	32000	5	8	
Dishi	55000	6	7	
Abhishek	92000	8	3	
Avantika	120000	15	1	
Ayushi	65750	7	5	

The attributes salary and year_of_experience are on different scale and hence attribute salary can take high priority over attribute year_of_experience in the model.

Types of Normalisation :

1 Min-Max Normalization: In most cases, standardization is used feature-wise

$$\hat{X}[:, i] = \frac{X[:, i] - \min(X[:, i])}{\max(X[:, i]) - \min(X[:, i])}$$

2 Z-score normalization In this technique, values are normalized based on a mean and standard deviation of the data

$$v' = \frac{v - \bar{A}}{\sigma_A}$$

v' , v is new and old of each entry in data respectively. σ_A , A is the standard deviation and mean of A respectively.

standardization (or Z-score normalization) is that the features will be rescaled so that they'll have the properties of a standard normal distribution with

$\mu=0$ and $\sigma=1$ where μ is the mean (average) and σ is the standard deviation from the mean; standard scores (also called z scores) of the samples are calculated as follows:

$$z=(x-\mu)/\sigma$$

Q20: What is the difference between normalization and Standardization with example?

In ML, every practitioner knows that feature scaling is an important issue. The two most discussed scaling methods are **Normalization** and **Standardization**. Normalization typically means it rescales the values into a range of [0,1].

It is an alternative approach to Z-score normalization (or standardization) is the so-called Min-Max scaling (often also called “normalization” - a common cause for ambiguities). In this approach, the data is scaled to a fixed range - **usually 0 to 1**. Scikit-Learn provides a transformer called **MinMaxScaler** for this. A Min-Max scaling is typically done via the following equation:

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

Example with sample data: Before Normalization: Attribute Price in Dollars Storage Space Camera

- Attribute Price in Dollars Storage Space Camera
- Mobile 1 250 16 12
- Mobile 2 200 16 8
- Mobile 3 300 32 16
- Mobile 4 275 32 8
- Mobile 5 225 16 16

After Normalization: (Values ranges from 0-1 which is working as expected)

- Attribute Price in Dollars Storage Space Camera
- Mobile 1 0.5 0 0.5
- Mobile 2 0 0 0
- Mobile 3 1 1 1
- Mobile 4 0.75 1 0
- Mobile 5 0.25 0 1

Standardization (or Z-score normalization) typically means rescales data to have a mean of 0 and a standard deviation of 1 (unit variance) Formula: $Z \text{ or } X_{\text{new}} = \frac{(x - \mu)}{\sigma}$ where μ is the mean (average), and σ is the standard deviation from the mean; standard scores (also called z scores) Scikit-Learn provides a transformer called StandardScaler for standardization **Example:** Let's take an approximately normally distributed set of numbers: 1, 2, 2, 3, 3, 3, 4, 4, and 5. Its mean is 3, and its standard deviation: 1.22. Now, let's subtract the mean from all data points. we get a new data set of: -2, -1, -1, 0, 0, 0, 1, 1, and 2. Now, let's divide each data point by 1.22. As you can see in the picture below, we get: -1.6, -0.82, -0.82, 0, 0, 0, 0.82, 0.82, and 1.63

Q1. What is upsampling and downsampling with examples?

The classification data set with skewed class proportions is called an imbalanced data set. Classes which make up a large proportion of the data sets are called majority classes. Those make up smaller proportions are minority classes.

Degree of imbalance Proportion of Minority Class

- 1>> Mild 20-40% of the data set
- 2>> Moderate 1-20% of the data set
- 3>> Extreme <1% of the data set

If we have an imbalanced data set, first try training on the true distribution. If the model works well and generalises, you are done! If not, try the following up sampling and down sampling technique.

1. Up-sampling

Upsampling is the process of randomly duplicating observations from the minority class to reinforce its signal.

First, we will import the resampling module from Scikit-Learn:

Module for resampling Python

1- From sklearn.utils import resample

Next, we will create a new Data Frame with an up-sampled minority class. Here are the steps:

1- First, we will separate observations from each class into different Data Frames.

2- Next, we will resample the minority class with replacement, setting the number of samples to match that of the majority class.

3- Finally, we'll combine the up-sampled minority class Data Frame with the original majority class Data Frame.

2-Down-sampling

Downsampling involves randomly removing observations from the majority class to prevent its signal from dominating the learning algorithm.

The process is similar to that of sampling. Here are the steps:

1-First, we will separate observations from each class into different Data Frames.

2-Next, we will resample the majority class without replacement, setting the number of samples to match that of the minority class.

3-Finally, we will combine the down-sampled majority class Data Frame with the original minority class Data Frame.

Q2. What is the statistical test for data validation with an example,

Chi-square, ANOVA test, Z statics, T statics, F statics,

Hypothesis Testing?

Before discussing the different statistical test, we need to get a clear understanding of what a null hypothesis is. A null hypothesis proposes that has no significant difference exists in the set of a given observation.

Null: Two samples mean are equal. **Alternate:** Two samples mean are not equal.

For rejecting the null hypothesis, a test is calculated. Then the test statistic is compared with a critical value, and if found to be greater than the critical value, the hypothesis will be rejected.

Critical Value:-

Critical values are the point beyond which we reject the null hypothesis. Critical value tells us, what is the probability of N number of samples, belonging to the same distribution. Higher, the critical value which means lower the probability of N number of samples belonging to the same distribution.

Critical values can be used to do hypothesis testing in the following way.

1. Calculate test statistic
2. Calculate critical values based on the significance level alpha
3. Compare test statistics with critical values.

IMP-If the test statistic is lower than the critical value, accept the hypothesis or else reject the hypothesis.

Chi-Square Test:-

A chi-square test is used if there is a relationship between two categorical variables.

Chi-Square test is used to determine whether there is a significant difference between the expected frequency and the observed frequency in one or more categories. Chi-square is also called the non-parametric test as it will not use any parameter

2-Anova test:-

ANOVA, also called an analysis of variance, is used to compare multiples (three or more) samples with a single test.

Useful when there are more than three populations. Anova compares the variance within and between the groups of the population. If the variation is much larger than the within variation, the means of different samples will not be equal. If the between and within variations are approximately the same size, then there will be no significant difference between sample means. Assumptions of ANOVA: 1-All populations involved follow a normal distribution. 2-All populations have the same variance (or standard deviation). 3-The samples are randomly selected and independent of one another.

ANOVA uses the mean of the samples or the population to reject or support the null hypothesis. Hence it is called parametric testing.

3-Z Statics:-

In a z-test, the samples are assumed to be normal distributed. A z score is calculated with population parameters as “population mean” and “population standard deviation” and it is used to validate a hypothesis that the sample drawn belongs to the same population.

The statistics used for this hypothesis testing is called z-statistic, the score for which is calculated as $z = (x - \mu) / (\sigma / \sqrt{n})$, where x = sample mean μ = population mean σ / \sqrt{n} = population standard deviation If the test statistic is lower than the critical value, accept the hypothesis or else reject the hypothesis

4- T Statics:-

A t-test used to compare the mean of the given samples. Like z-test, t-test also assumed a normal distribution of the samples. A t-test is used when the population parameters (mean and standard deviation) are unknown.

There are three versions of t-test

1. Independent samples t-test which compare means for two groups
2. Paired sample t-test which compares mean from the same group at different times
3. Sample t-test, which tests the mean of the single group against the known mean. The statistic for hypothesis testing is called t-statistic, the score for which is calculated as $t = (x_1 - x_2) / (\sigma / \sqrt{n_1} + \sigma / \sqrt{n_2})$, where

x_1 = mean of sample A, x_2 = mean of sample B,

n_1 = size of sample 1 n_2 = size of sample 2

5- F Statistics:-

The F-test is designed to test if the two population variances are equal. It compares the ratio of the two variances. Therefore, if the variances are equal, then the ratio of the variances will be 1.

The F-distribution is the ratio of two independent chi-square variables divided by their respective degrees of freedom.

$F = s_1^2 / s_2^2$ and where $s_1^2 > s_2^2$.

If the null hypothesis is true, then the F test-statistic given above can be simplified. This ratio of sample variances will be tested statistic used. If the null hypothesis is false, then we will reject the null hypothesis that the ratio was equal to 1 and our assumption that they were equal.

Q3. What is the Central limit theorem?

Central Limit Theorem

Definition: The theorem states that as the size of the sample increases, the distribution of the mean across multiple samples will approximate a Gaussian distribution (Normal). Generally, sample sizes equal to or greater than 30 are considered sufficient for the CLT to hold. It means that the distribution of the sample means is normally distributed. The average of the

sample means will be equal to the population mean. This is the key aspect of the theorem.

Assumptions:

1. The data must follow the randomization condition. It must be sampled randomly
2. Samples should be independent of each other. One sample should not influence the other samples
3. Sample size should be no more than 10% of the population when sampling is done without replacement
4. The sample size should be sufficiently large. The mean of the sample means is denoted as:

$$\mu \bar{X} = \mu$$

Where,

$\mu \bar{X}$ = Mean of the sample means μ = Population mean and, the standard deviation of the sample mean is denoted as:

$$\sigma \bar{X} = \sigma / \sqrt{n}$$

Where,

$\sigma \bar{X}$ = Standard deviation of the sample mean σ = Population standard deviation n = sample size

A sufficiently large sample size can predict the characteristics of a population accurately. For Example, we shall take a uniformly distributed data:

Randomly distributed data: Even for a randomly (Exponential) distributed data the plot of the means is normally distributed.

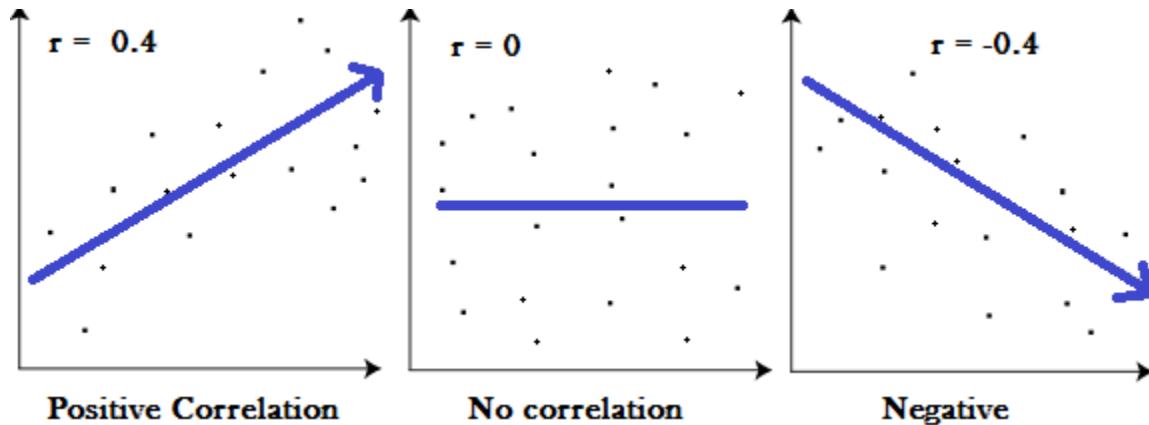
The advantage of CLT is that we need not worry about the actual data since the means of it will always be normally distributed. With this, we can create component intervals, perform T-tests and ANOVA tests from the given samples.

Q4. What is the correlation and coefficient?

What is the Correlation Coefficient?

The correlation coefficient is a statistical measure that calculates the strength of the relationship between the relative movements of two

variables. We use it to measure both the strength and direction of a linear relationship between two variables the values range between -1.0 and 1.0. A calculated number greater than 1.0 or less than -1.0 means that there was an error in the correlation measurement. A correlation of -1.0 shows a perfect negative correlation, while a correlation of 1.0 shows a perfect positive correlation.



Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:

1 indicates a strong positive relationship. -1 indicates a strong negative relationship. A result of zero indicates no relationship at all.

Meaning

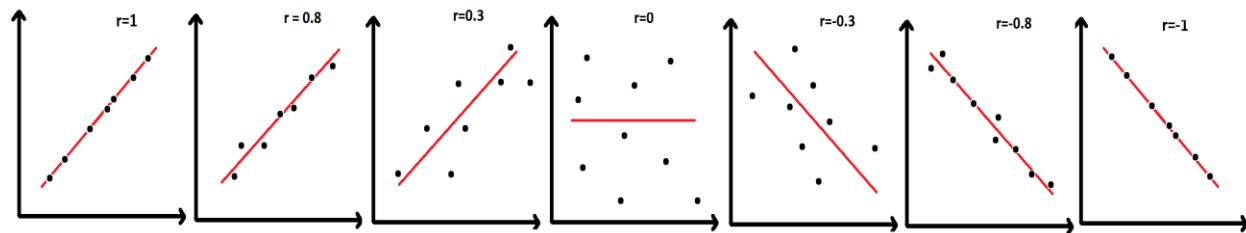
1. A correlation coefficient of 1 means that for every positive increase in one variable, there is a positive increase in a fixed proportion in the other. For example, shoe sizes go up in (almost) perfect correlation with foot length.
2. A correlation coefficient of -1 means that for every positive increase in one variable, there is a negative decrease of a fixed proportion in the other. For example, the amount of gas in a tank decreases in (almost) perfect correlation with speed.
3. Zero means that for every increase, there isn't a positive or negative increase. The two just aren't related.

What is a Negative Correlation?

Negative correlation is a relationship between two variables in which one variable increases as the other decreases, and vice versa. In statistics, a perfect negative correlation is represented by the value -1. Negative correlation or inverse correlation is a relationship between two variables whereby they move in opposite directions. If variables X and Y have a negative correlation (or are negatively correlated), as X increases in value, Y will decrease; similarly, if X decreases in value, Y will increase.

What Is Positive Correlation?

Positive correlation is a relationship between two variables in which both variables move in tandem—that is, in the same direction. A positive correlation exists when one variable decreases as the other variable decreases or one variable increases while the other increases.



We use the correlation coefficient to measure the strength and direction of the linear relationship between two numerical variables X and Y. The correlation coefficient for a sample of data is denoted by r .

Pearson Correlation Coefficient

Pearson is the most widely used correlation coefficient. Pearson correlation measures the linear association between continuous variables. In other words, this coefficient quantifies the degree to which a relationship between two variables can be described by a line. Formula developed by Karl Pearson over 120 years ago is still the most widely used today. The formula for the correlation (r) is

Correlation Coefficient Formula

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}}$$

Where n is the number of pairs of data;

Are the sample means of all the x-values and all the y-values, respectively; and sx and sy are the sample standard deviations of all the x- and y-values, respectively.

1. Find the mean of all the x-values and mean of all y-values.
2. Find the standard deviation of all the x-values (call it sx) and the standard deviation of all the y-values (call it sy). For example, to find sx, you would use the following equation:
3. For each of the n pairs (x, y) in the data set, take
4. Add up the n results from Step 3.
5. Divide the sum by sx * sy.
6. Divide the result by n – 1, where n is the number of (x, y) pairs. (It's the same as multiplying by 1 over n – 1.) This gives you the correlation, r.

Q5: What is the difference between machine learning and deep learning?

Machine Learning | deep learning

Machine Learning is a technique to learn from that data and then apply what has been learnt to make an informed decision | The main difference between deep and machine learning is, machine learning models become better progressively but the model still needs some guidance. If a machine-learning model returns an inaccurate prediction then the programmer needs to fix that problem explicitly but in the case of deep learning, the model does it by himself.

>Machine Learning can perform well with small size data also | Deep Learning does not perform as good with smaller datasets.

>Machine learning can work on some low-end machines also | Deep Learning involves many matrix multiplication operations which are better suited for GPUs

>Features need to be identified and extracted as per the domain before pushing them to the algorithm | Deep learning algorithms try to learn high-level features from data.

>It is generally recommended to break the problem into smaller chunks, solve them and then combine the results | It generally focusses on solving the problem end to end

>Training time is comparatively less | Training time is comparatively more

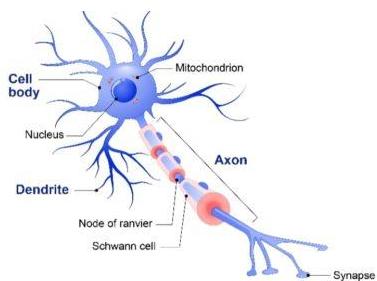
>Results are more interpretable | Results Maybe more accurate but less interpretable

> No use of Neural networks | uses neural networks

> Solves comparatively less complex problems | Solves more complex problems.

Q6: What is perceptron and how it is related to human neurons?

If we focus on the structure of a biological neuron, it has dendrites, which are used to receive inputs. These inputs are summed in the cell body and using the Axon it is passed on to the next biological neuron as shown below.

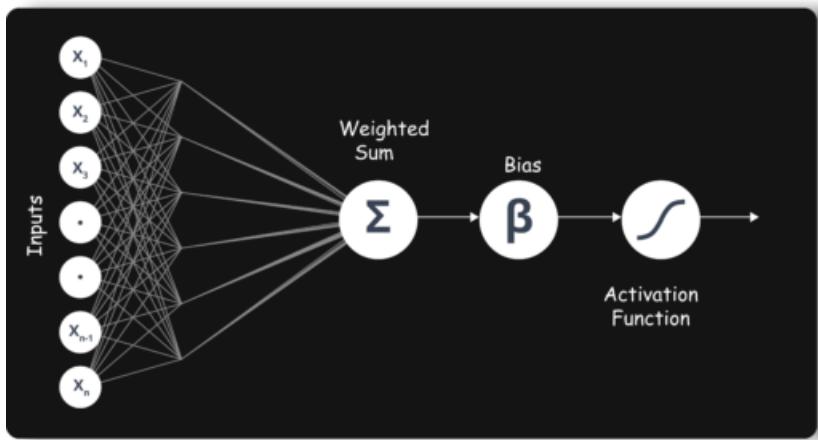


Dendrite: Receives signals from other neurons

Cell Body: Sums all the inputs

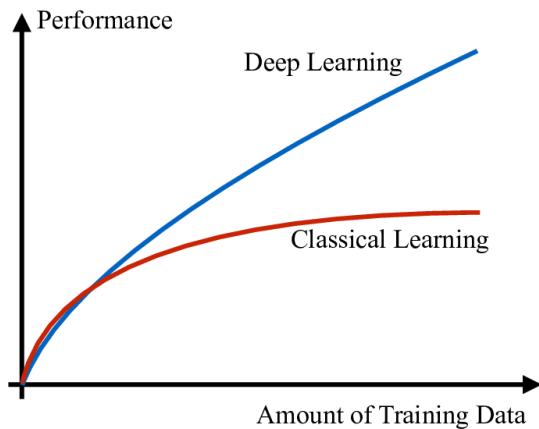
Axon: It is used to transmit signals to the other cells

Similarly, a perceptron receives multiple inputs, applies various transformations and functions and provides an output. A Perceptron is a linear model used for binary classification. It models a neuron, which has a set of inputs, each of which is given a specific weight. The neuron computes some function on these weighted inputs and gives the output.



Q7: Why deep learning is better than machine learning?

Though traditional ML algorithms solve a lot of our cases, they are not useful while working with high dimensional data that is where we have a large number of inputs and outputs. For example, in the case of handwriting recognition, we have a large amount of input where we will have different types of inputs associated with different types of handwriting.



The second major challenge is to tell the computer what are the features it should look for that will play an important role in predicting the outcome as well as to achieve better accuracy while doing so.

Q8: What kind of problem can be solved by using deep learning?

Deep Learning is a branch of Machine Learning, which is used to solve problems in a way that mimics the human way of solving problems.

Examples:

- Image recognition
- Object Detection
- Natural Language processing- Translation, Sentence formations, text to speech, speech to text
- understand the semantics of actions

Q9: List down all the activation function using mathematical

Expression and example. What is the activation function?

Activation functions are very important for an Artificial Neural Network to learn and make sense of something complicated and the Non-linear complex functional mappings between the inputs and response variable. They introduce non-linear properties to our Network. Their main purposes are to convert an input signal of a node in an A-NN to an output signal.

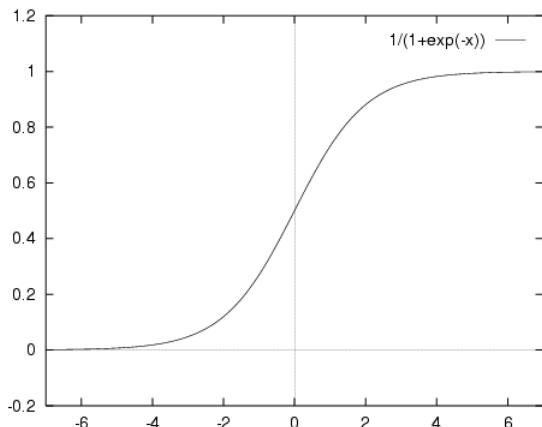
So why do we need Non-Linearities?

Non-linear functions are those, which have a degree more than one, and they have a curvature when we plot a Non-Linear function. Now we need a Neural Network Model to learn and represent almost anything and any arbitrary complex function, which maps inputs to outputs. Neural-Networks are considered Universal Function Approximations. It means that they can compute and learn any function at all.

Most popular types of Activation functions -

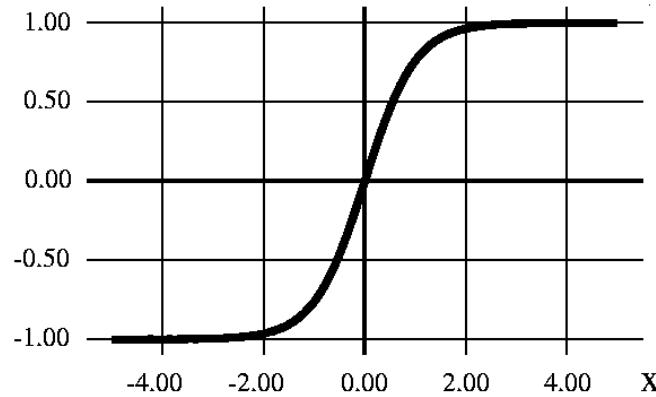
- Sigmoid or Logistic
- Tanh — Hyperbolic tangent
- ReLu -Rectified linear units

Sigmoid Activation function: It is a activation function of form $f(x) = 1 / (1 + \exp(-x))$. Its Range is between 0 and 1. It is an S-shaped curve. It is easy to understand.

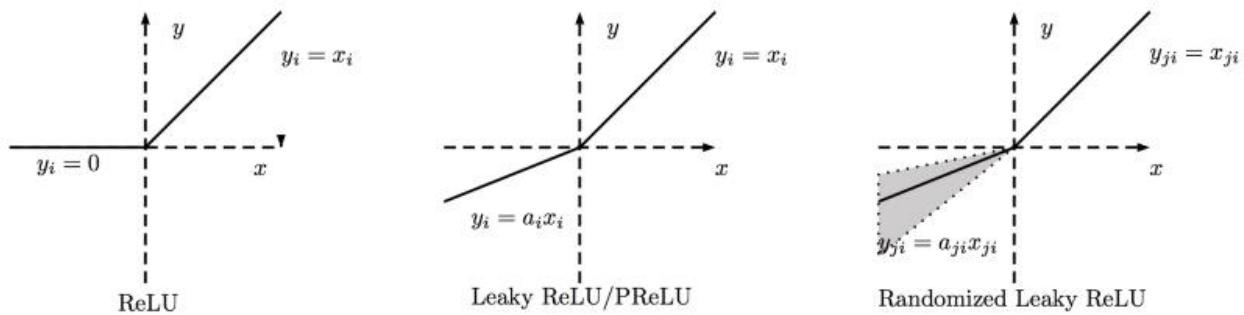


Hyperbolic Tangent function- Tanh : It's mathematical formula is $f(x) = \tanh(x) = (\exp(x) - \exp(-x)) / (\exp(x) + \exp(-x))$. Now it's the output is zero centred because its range in between -1 to 1 i.e. $-1 < \text{output} < 1$. Hence optimisation is easier in this method; Hence in practice, it is always preferred over Sigmoid function.

hyperbolic tangent function

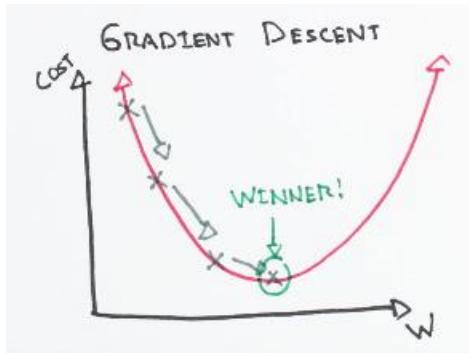


ReLU- Rectified Linear units: It has become more popular in the past couple of years. It was recently proved that it has six times improvement in convergence from Tanh function. It's $R(x) = \max(0, x)$ i.e. if $x < 0$, $R(x) = 0$ and if $x \geq 0$, $R(x) = x$. Hence as seen that mathematical form of this function, we can see that it is very simple and efficient. Many times in Machine learning and computer science we notice that most simple and consistent techniques and methods are only preferred and are the best. Hence, it avoids and rectifies the vanishing gradient problem. Almost all the deep learning Models use ReLU nowadays.



Q10: Detail explanation about gradient decent using example and Mathematical expression?

Gradient descent is an optimisation algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by negative of the gradient. In machine learning, we used gradient descent to update the parameters of our model. Parameters refer to coefficients in the Linear Regression and weights in neural networks.



The size of these steps called the learning rate. With the high learning rate, we can cover more ground each step, but we risk overshooting the lower point since the slope of the hill is constantly changing. With a very lower learning rate, we can confidently move in the direction of the negative gradient because we are recalculating it so frequently. The Lower learning rate is more precise, but calculating the gradient is time-consuming, so it will take a very large time to get to the bottom.

Math

Now let's run gradient descent using new cost function. There are two parameters in cost function we can control: m (weight) and b (bias). Since we need to consider that the impact each one has on the final prediction, we need to use partial derivatives. We calculate the partial derivative of the cost function concerning each parameter and store the results in a gradient.

Math

Given the cost function:

$$f(m, b) = \frac{1}{N} \sum_{i=1}^n (y_i - (mx_i + b))^2$$

The gradient can be calculated as:

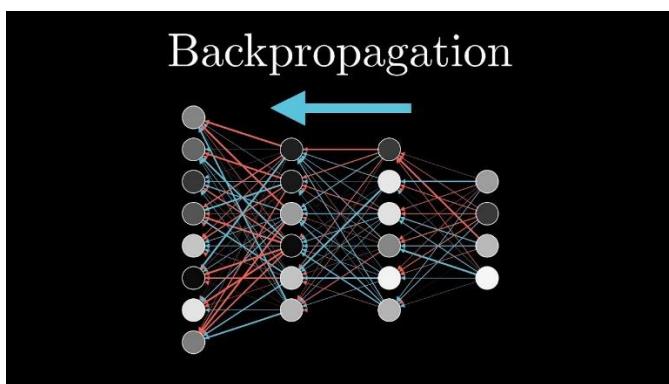
$$f'(m, b) = \begin{bmatrix} \frac{df}{dm} \\ \frac{df}{db} \end{bmatrix} = \begin{bmatrix} \frac{1}{N} \sum -2x_i(y_i - (mx_i + b)) \\ \frac{1}{N} \sum -2(y_i - (mx_i + b)) \end{bmatrix}$$

To solve for the gradient, we iterate by our data points using our new m and b values and compute the partial derivatives. This new gradient tells us about the slope of the cost function at our current position (current parameter values) and the directions we should move to update our parameters. The learning rate controls the size of our update.

Q11: What is backward propagation?

Back-propagation is the essence of the neural net training and this method of fine-tuning the weights of a neural net based on the errors rate obtained in the previous epoch. Proper tuning of the weights allows us to reduce error rates and to make the model reliable by increasing its generalisation.

Backpropagation is a short form of "backward propagation of errors." This is the standard method of training artificial neural networks. This helps to calculate the gradient of a loss function with respects to all the weights in the network.



Most prominent advantages of Backpropagation are:

- Backpropagation is the fast, simple and easy to program.
- It has no parameters to tune apart from the numbers of input.
- It is the flexible method as it does not require prior knowledge about the network
- It is the standard method that generally works well.
- It does not need any special mentions of the features of the function to be learned.

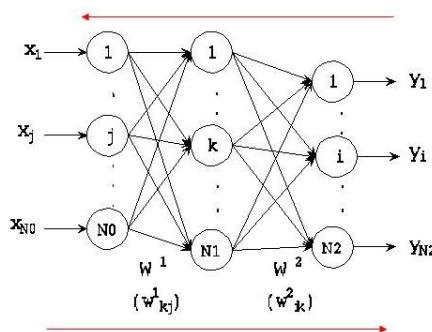
The BackPropagation Algorithm

Main idea:

For each example in the training set:

- compute the output signal
- compute the error corresponding to the output level
- propagate the error back into the network and store the corresponding delta values for each layer
- adjust each weight by using the error signal and input signal for each layer

Computation of the error signal (BACKWARD)



Computation of the output signal (FORWARD)

Q12: How we assign weights in deep learning?

We already know that in a neural network, weights are usually initialised randomly and that kind of initialisation takes a fair/significant amount of repetitions to converge to the least loss and reach the ideal weight matrix. The problem is, that kind of initialisation is prone to vanishing or exploding gradient problems.

General ways to make it initialise better weights:

ReLU activation function in the deep nets.

1. Generate a random sample of weights from a Gaussian distribution having mean 0 and a standard deviation of 1.
2. Multiply the sample with the square root of $(2/n_i)$. Where n_i is the number of input units for that layer.

b) Likewise, if you're using Tanh activation function :

1. Generate a random sample of weights from a Gaussian distribution having mean 0 and a standard deviation of 1.
2. Multiply the sample with the square root of $(1/n_i)$ where n_i is several input units for that layer.

Q13: What is optimiser in deep learning, and which one is the best?

Deep learning is an iterative process. With so many hyperparameters to tune or methods to try, it is important to be able to train models fast, to quickly complete the iterative cycle. This is the key to increase the speed and efficiency of a machine learning team.

Hence the importance of optimisation algorithms such as stochastic gradient descent, min-batch gradient descent, gradient descent with momentum and the Adam optimiser.

Adam optimiser is the best one.

Given an algorithm $f(x)$, it helps in either minimisation or maximisation of the value of $f(x)$. In this context of deep learning, we use optimisation algorithms to train the neural network by optimising the cost function J .

The cost function is defined as:

$$J(W, b) = \sum_{i=1}^m L(y'^i, y^i)$$

The value of the cost function J is the mean of the loss L between the predicted value y' and actual value y . The value y'' is obtained during the forward propagation step and makes use of the Weights W and biases b of the network. With the help of optimisation algorithms, we minimise the value of Cost Function J by updating the values of trainable parameters W and b .

Q14: What is gradient descent, mini-batch gradient descent, batch gradient decent, stochastic gradient decent and adam?

Gradient Descent

it is an iterative machine learning optimisation algorithm to reduce the cost function, and help models to make accurate predictions.

Gradient indicates the direction of increase. As we want to find the minimum points in the valley, we need to go in the opposite direction of the gradient. We update the parameters in the negative gradient direction to minimise the loss.

$$\theta = \theta - \eta \nabla J(\theta; x, y)$$

Where θ is the weight parameter, η is the learning rate, and $\nabla J(\theta; x, y)$ is the gradient of weight parameter θ

Types of Gradient Descent

Different types of Gradient descents are

- Batch Gradient Descent or Vanilla Gradient Descent
- Stochastic Gradient Descent

- Mini batch Gradient Descent

Batch Gradient Descent

In the batch gradient, we use the entire dataset to compute the gradient of the cost function for each iteration for gradient descent and then update the weights.

Stochastic Gradient descent

Stochastic gradient descent, we use a single data point or example to calculate the gradient and update the weights with every iteration.

We first need to shuffle the datasets so that we get a completely randomised dataset. As the datasets are random and weights, are updated for every single example, an update of the weights and the cost functions will be noisy jumping all over the place

Mini Batch Gradient descent

Mini-batch gradients is a variation of stochastic gradient descent where instead of a single training example, a mini-batch of samples are used.

Mini -batch gradient descent is widely used and converges faster and is more stable.

The batch size can vary depending upon the dataset.

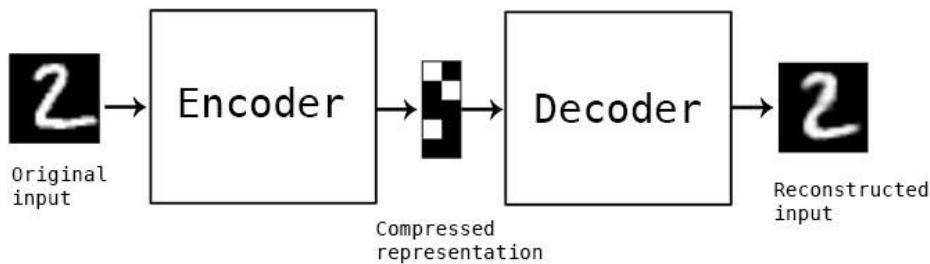
As we take batches with different samples, it reduces the noise which is a variance of the weights updates, and that helps to have a more stable converge faster.

Q15: What are autoencoders?

An **autoencoder**, neural networks that have three layers:

An input layer, a hidden layer which is also known as encoding layer, and a decoding layer. This network is trained to reconstruct its inputs, which forces the hidden layer to try to learn good representations of the inputs.

An autoencoder neural network is an unsupervised Machine-learning algorithm that applies backpropagation, setting the target values to be equal to the inputs. An autoencoder is trained to attempt to copy its input to its output. Internally, it has a hidden layer which describes a code used to represent the input.



Autoencoder Components:

Autoencoders consist of 4 main parts:

- 1- Encoder: In this, the model learns how to reduce the input dimensions and compress the input data into an encoded representation.
- 2- Bottleneck: In this, the layer that contains the compressed representation of the input data. This is the lowest possible dimension of the input data.
- 3- Decoder: In this, the model learns how to reconstruct the data from the encoded representation to be as close to the original inputs as possible.
- 4- Reconstruction Loss: In this method that measures how well the decoder is performing and how close the output is related to the original input.

Types of Autoencoders :

1. Denoising auto encoder
2. Sparse auto encoder
3. Variational auto encoder (VAE)
4. Contractive auto encoder (CAE)

Q16: What is CNN?

This is the simple application of a filter to an input that results in inactivation. Repeated application of the same filter to input results in a map of activations called a feature map, indicating the locations and strength of a detected feature in input, such as an image.

Convolutional layers are the major building blocks which are used in convolutional neural networks.

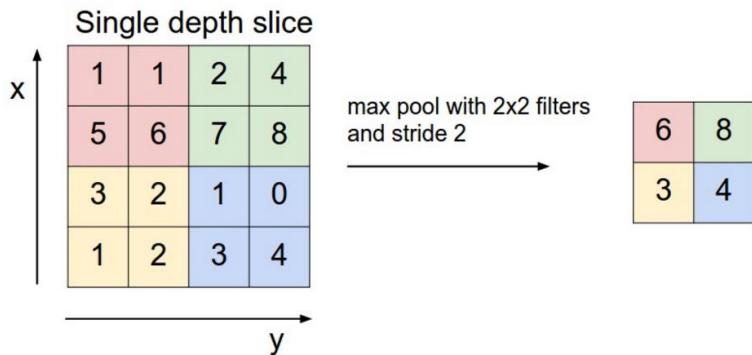
A covnets is the sequence of layers, and every layer transforms one volume to another through differentiable functions.

Different types of layers in CNN:

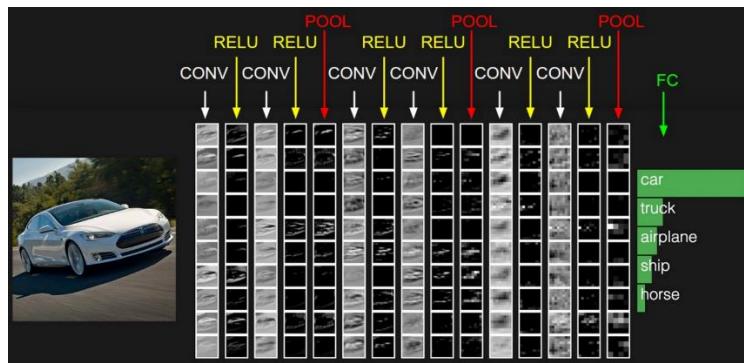
Let's take an example by running a covnets on of image of dimensions **32 x 32 x 3**.

1. Input Layer: It holds the raw input of image with width 32, height 32 and depth 3.
2. Convolution Layer: It computes the output volume by computing dot products between all filters and image patches. Suppose we use a total of 12 filters for this layer we'll get output volume of dimension 32 x 32 x 12.
3. Activation Function Layer: This layer will apply the element-wise activation function to the output of the convolution layer. Some activation functions are RELU: $\max(0, x)$, Sigmoid: $1/(1+e^{-x})$, Tanh, Leaky RELU, etc. So the volume remains unchanged. Hence output volume will have dimensions 32 x 32 x 12.
4. Pool Layer: This layer is periodically inserted within the covnets, and its main function is to reduce the size of volume which makes the

computation fast reduces memory and also prevents overfitting. Two common types of pooling layers are max pooling and average pooling. If we use a max pool with 2×2 filters and stride 2, the resultant volume will be of dimension $16 \times 16 \times 12$.



5. Fully-Connected Layer: This layer is a regular neural network layer that takes input from the previous layer and computes the class scores and outputs the 1-D array of size equal to the number of classes.

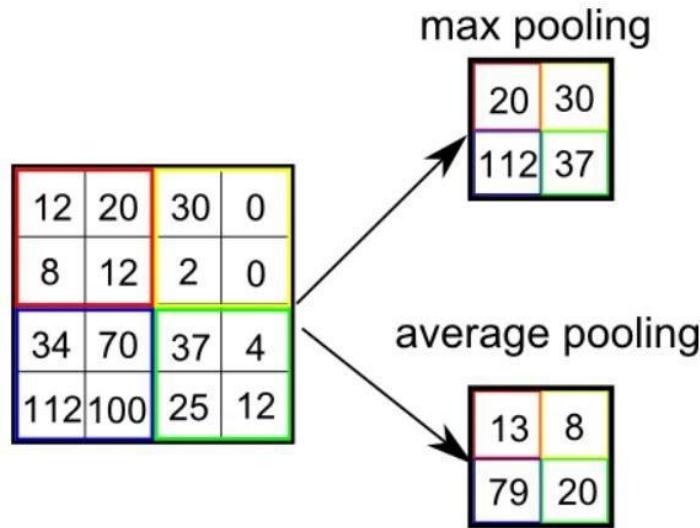


Q17: What is pooling, padding, filtering operations on CNN?

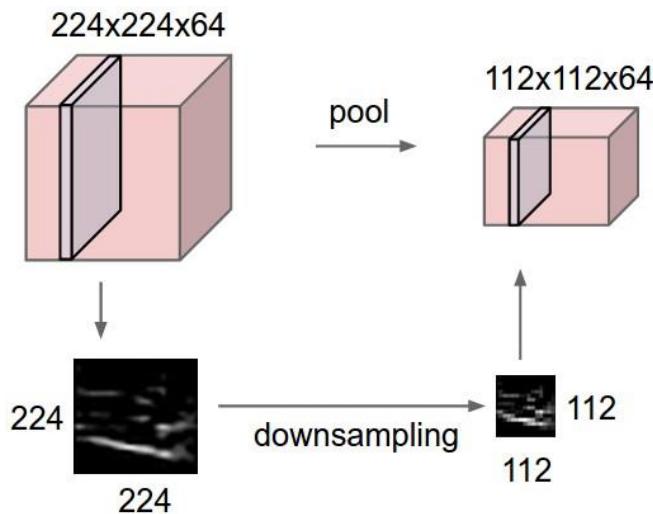
Pooling Layer

It is commonly used to periodically insert a Pooling layer in-between successive Conv layers in a ConvNet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also

control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation.

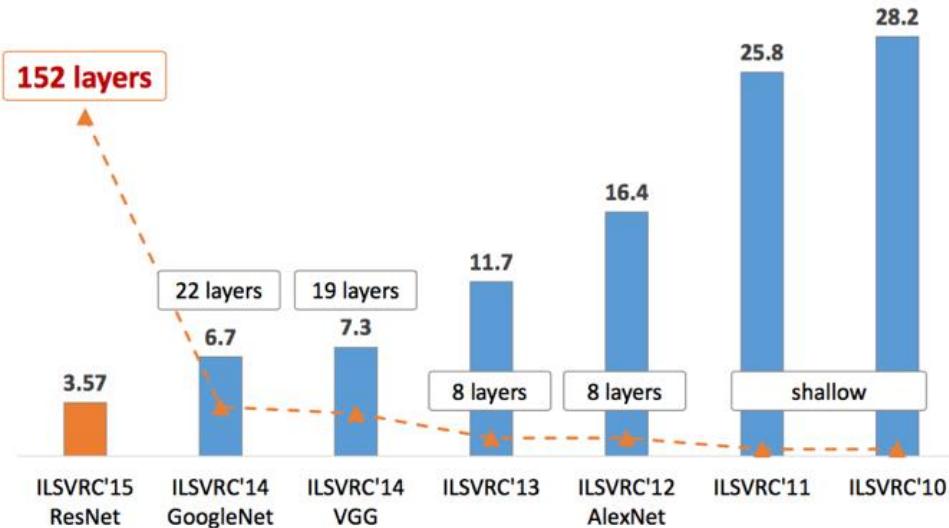


The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by two along both width and height, discarding 75% of the activations. Every MAX operation would, in this case, be taking a max over four numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged.



Q18: What is the Evolution technique of CNN?

It all started with LeNet in 1998 and eventually, after nearly 15 years, lead to groundbreaking models winning the ImageNet Large Scale Visual Recognition Challenge which includes AlexNet in 2012 to Google Net in 2014 to ResNet in 2015 to an ensemble of previous models in 2016. In the last two years, no significant progress has been made, and the new models are an ensemble of previous groundbreaking models.



LeNet in 1998

LeNet is a 7-level convolutional network by LeCun in 1998 that classifies digits and used by several banks to recognise the hand-written numbers on cheques digitised in 32x32 pixel greyscale input images.

AlexNet in 2012

AlexNet: It is considered to be the first paper/ model, which rose the interest in CNNs when it won the ImageNet challenge in the year 2012. It is a deep CNN trained on ImageNet and outperformed all the entries that year.

VGG in 2014

VGG was submitted in the year 2013, and it became a runner up in the ImageNet contest in 2014. It is widely used as a simple architecture compared to AlexNet.

GoogleNet in 2014

In 2014, several great models were developed like VGG, but the winner of the ImageNet contest was GoogleNet.

GoogLeNet proposed a module called the inception modules that includes skipping connections in the network, forming a mini-module, and this module is repeated throughout the network.

ResNet in 2015

There are 152 layers in the Microsoft ResNet. The authors showed empirically that if you keep on adding layers, the error rate should keep on decreasing in contrast to “plain nets” we’re adding a few layers resulted in higher training and test errors.

Q19: How to initialise biases in deep learning?

It is possible and common to initialise the biases to be zero since the random numbers in the weights provide the asymmetry breaking. For ReLU non-linearities, some people like to use small constant value such as 0.01 for all biases because this ensures that all ReLU units fire in the beginning, therefore obtain, and propagate some gradient. However, it is unclear if this provides a consistent improvement (in fact some results seem to indicates that this performs worst) and it is more commonly used to use 0 bias initialisation.

Q20: What is learning Rate?

Learning Rate

The learning rate controls how much we should adjust the weights concerning the loss gradient. Learning rates are randomly initialised.

Lower the values of the learning rate slower will be the convergence to global minima.

Higher values for the learning rate will not allow the gradient descent to converge

Since our goal is to minimise the function cost to find the optimised value for weights, we run multiples iteration with different weights and calculate the cost to arrive at a minimum cost

Q1: What are Epochs?

One Epoch is an ENTIRE dataset is passed forwards and backwards through the neural network.

Since one epoch is too large to feed to the computer at once, we divide it into several smaller batches.

We always use more than one Epoch because *one epoch leads to underfitting*.

As the number of epochs increases, several times the weight are changed in the neural network and the curve goes from underfitting up to optimal to overfitting curve.

Q2. What is the batch size?

Batch Size

The total number of training and examples present in a single batch.

Unlike the learning rate hyperparameter where its value doesn't affect computational time, the batch sizes must be examined in conjunctions with the execution time of training. The batch size is limited by hardware's memory, while the learning rate is not. Leslie recommends using a batch size that fits in hardware's memory and enables using larger learning rate.

If our server has multiple GPUs, the total batch size is the batch size on a GPU multiplied by the numbers of GPU. If the architectures are small or your hardware permits very large batch sizes, then you might compare the performance of different batch sizes. Also, recall that small batch sizes add regularization while large batch sizes add less, so utilize this while balancing the proper amount of regularization. It is often better to use large batch sizes so a larger learning rate can be used.

Q3: What is dropout in Neural network?

Dropout refers to ignoring units during the training phase of a certain set of neurons which is chosen randomly. These units are not considered during the particular forward or backward pass.

More technically, at each training stage, individual nodes are either dropped out of the net with probability $1-p$ or kept with probability p , so that a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed.

We need Dropout *to prevent over-fitting*

A dropout is an approach to regularization in neural networks which helps to reduce interdependent learning amongst the neurons.

Where to use

Dropout is implemented per-layer in a neural network.

It can be used with most types of layers, such as dense fully connected layers, convolutional layers, and recurrent layers such as the long short-term memory network layer.

Dropout may be implemented on any or all hidden layers in the network as well as the visible or input layer. It is not used on the output layer.

Benefits:-

1. Dropout forces a neural network to learn more robust features that are very useful in conjunction with different random subsets of the other neurons.
2. Dropout generally doubles the number of iterations required to converge. However, the training time for each epoch is less.

Q4: List down hyperparameter tuning in deep learning.

The process of setting the hyper-parameters requires expertise and extensive trial and error. There are no simple and easy ways to set hyper-parameters — specifically, learning rate, batch size, momentum, and weight decay.

Approaches to searching for the best configuration:

- Grid Search
- Random Search

Approach

1. Observe and understand the clues available during training by monitoring validation/test loss early in training, tune your architecture and hyper-parameters with short runs of a few epochs.
2. Signs of *underfitting* or *overfitting* of the test or validation loss early in the training process are useful for tuning the hyper-parameters.

Tools for Optimizing Hyperparameters

- Sage Maker
- Comet.ml
- Weights & Biases
- Deep Cognition
- Azure ML

Q5: What do you understand by activation function and error functions?

Error functions

In most learning networks, an error is calculated as the difference between the predicted output and the actual output.

$$J(w) = p - \hat{p}$$

The function that is used to compute this error is known as Loss Function $J(\cdot)$. Different loss functions will give different errors for the same prediction, and thus have a considerable effect on the performance of the model. One of the most widely used loss function is mean square error, which calculates the square of the difference between the actual values and predicted value. Different loss functions are used to deals with a different type of tasks, i.e. regression and classification.

Regressive loss functions:

Mean Square Error

Absolute error

Smooth Absolute Error

Classification loss functions:

1. Binary Cross-Entropy
2. Negative Log-Likelihood
3. Margin Classifier
4. Soft Margin Classifier

Activation functions decide whether a neuron should be activated or not by calculating a weighted sum and adding bias with it. The purpose of the activation function is to introduce non-linearity into the output of a neuron.

In a neural network, we would update the weights and biases of the neurons based on the error at the outputs. This process is known as back-propagation. Activation function makes the back-propagation possible since the gradients are supplied along with the errors to update the weights and biases.

Q6: Why do we need Non-linear activation functions?

A neural network without activation functions is essentially a linear regression model. The activation functions do the non-linear transformation to the input, making it capable of learning and performing more complex tasks.

1. Identity
2. Binary Step
3. Sigmoid
4. Tanh
5. ReLU
6. Leaky ReLU

7. Softmax

The activation functions do the non-linear transformation to the input, making it capable of learning and performing more complex tasks.

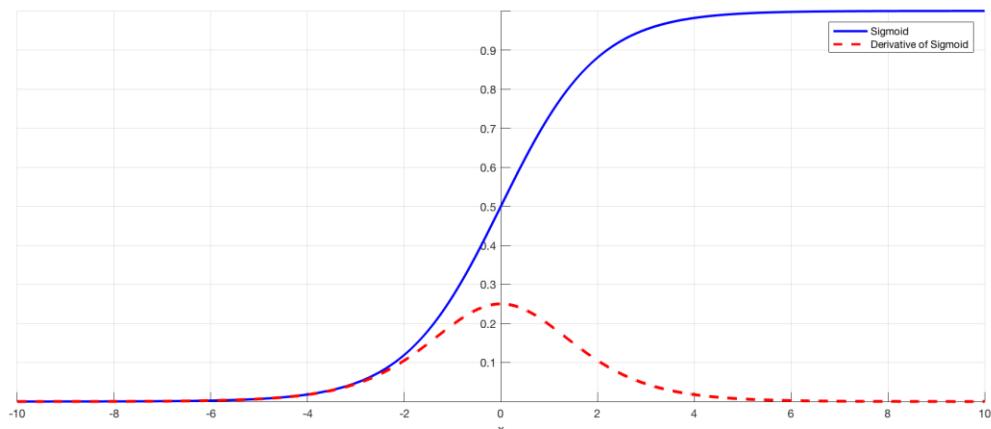
Q7: What do you understand by vanishing gradient problem and how can we solve that?

The problem:

As more layers using certain activation function are added to neural networks, the gradients of the loss function approach zero, making the networks tougher to train.

Why:

Certain activation functions, like the sigmoid function, squishes a large input space into a small input space between 0 and 1. Therefore, a large change in the input of the sigmoid function will cause a small change in the output. Hence, the derivative becomes small.



For shallow networks with only a few layers that use these activations, this isn't a big problem. However, when more layers are used, it can cause the gradient to be too small for training to work effectively.

However, when n hidden layers use an activation like the sigmoid function, n small derivatives are multiplied together. Thus, the gradient decreases exponentially as we propagate down to the initial layers.

Solutions:

The simplest solution is to use other activation functions, such as ReLU, which doesn't cause a small derivative.

Residual networks are another solution, as they provide residual connections straight to earlier layers.

Finally, batch normalization layers can also resolve the issue.

Q8: What is Transfer learning in deep learning ?

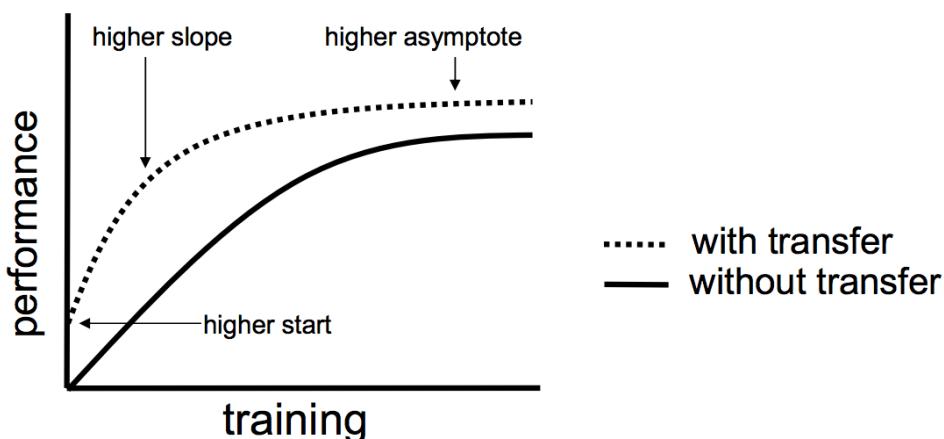
Transfer learning: It is a machine learning method where a model is developed for the task is again used as the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task.

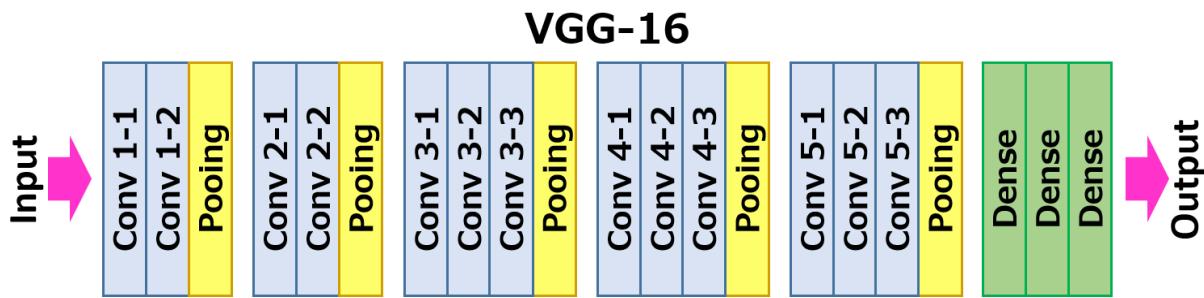
Transfer learning is an optimization that allows rapid progress or improved performance when modelling the second task.

Transfer learning only works in deep learning if the model features learned from the first task are general.



Q9: What is VGG16 and explain the architecture of VGG16?

VGG-16 is a simpler architecture model since it's not using many hyperparameters. It always uses 3×3 filters with the stride of 1 in convolution layer and uses SAME padding in pooling layers 2×2 with a stride of 2.

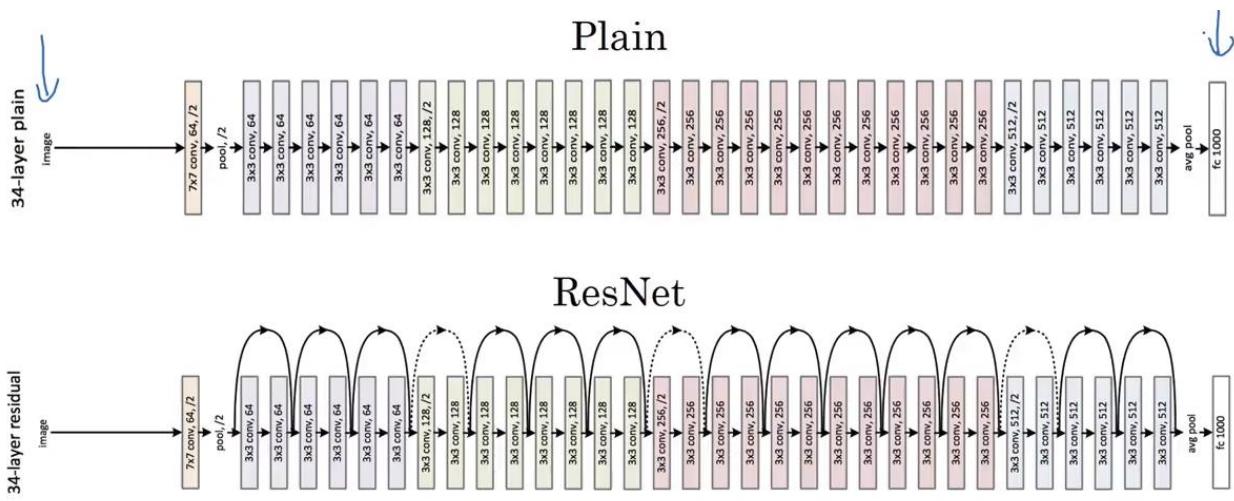


This architecture is from the VGG group, Oxford. It improves AlexNet by replacing the large kernel-sized filter with multiple 3×3 kernel-sized filters one after another. With a given receptive field (the effective area size of input image on which output depends), multiple stacked smaller size kernel is better than the one with a larger size kernel because multiple non-linear layers increases the depth of the network which enables it to learn more complex features, and that too at a lower cost.

Three fully connected layers follow the VGG convolutional layers. The width of the networks starts at the small value of 64 and increases by a factor of 2 after every sub-sampling/pooling layer. It achieves the top-5 accuracy of 92.3 % on ImageNet.

Q10: What is RESNET?

The winner of ILSRVC 2015, it also called as Residual Neural Network (ResNet) by Kaiming. This architecture introduced a concept called “skip connections”. Typically, the input matrix calculates in two linear transformations with ReLU activation function. In Residual network, it directly copies the input matrix to the second transformation output and sums the output in final ReLU function.



Skip Connection

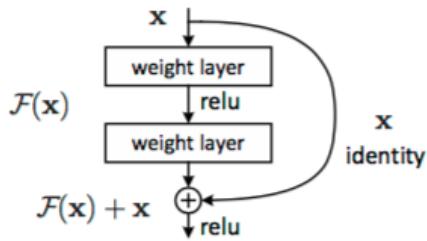


Figure 2. Residual learning: a building block.

Experiments in paper four can judge the power of the residual network. The plain 34 layer network had high validation error than the 18 layers plain network. This is where we realize the degradation problems. And the same 34 layers network when converted to the residual network has much less training error than the 18 layers residual network.

Q11: What is ImageNet?

ImageNet is a project aimed at (manually) labelling and categorizing images into almost 22,000 separate object categories for computer vision researches.

When we hear the about “*ImageNet*” in the context of deep learning and Convolutional Neural Network, we are referring to *ImageNet Large Scale Visual Recognition Challenge*.

The main aim of this image classification challenge is to train the model that can correctly classify an input image into the 1,000 separate objects category.

Models are trained on the ~1.2 million training images with another 50,000 images for validation and 100,000 images for testing.

These 1,000 image categories represent object classes that we encounter in our day-to-day lives, such as species of dogs, cats, various household objects, vehicle types, and much more.

When it comes to the image classification, the ImageNet challenge is the “de facto” benchmark for computer vision classification algorithms — and the leaderboard for this challenge has been dominated by Convolutional Neural Networks and Deep learning techniques since 2012.



Q12: What is DarkNet?

DarkNet is a framework used to train neural networks; it is open source and written in C/CUDA and serves as the basis for YOLO. Darknet is also used as the framework for training YOLO, meaning it sets the architecture of the network.

Clone the repo locally, and you have it. To compile it, run a make. But first, if you intend to use the GPU capability, you need to edit the **Makefile** in the first two lines, where you tell it to compile for GPU usage with CUDA drivers.

Q13: What is YOLO and explain the architecture of YOLO (you only

Look Once). One use case?

YOLO v1

The first YOLO You only look once (YOLO) version came about May 2016 and sets the core of the algorithm, the following versions are improvements that fix some drawbacks.

In short, YOLO is a network “inspired by” Google Net. It has 24 convolutional layers working as the feature extractors and two dense layers for making the predictions. The architecture works upon is called Darknet, a neural network framework created by the first author of the YOLO paper.

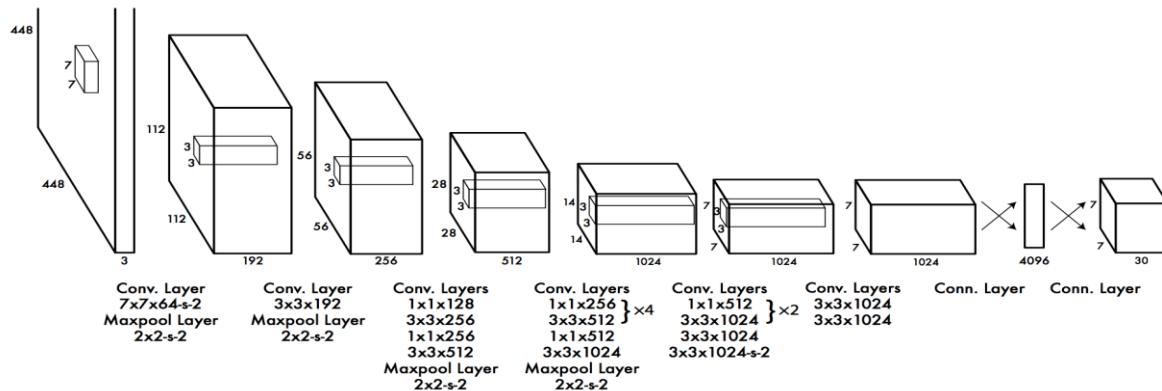
Core Concept:-

The algorithm works off by dividing the image into the grid of the cells, for each cell bounding boxes and their scores are predicted, alongside class probabilities. The confidence is given in terms of IOU (*intersection over union*), metric, which is measuring how much the detected object overlaps with the ground truth as a fraction of the total area spanned by the two together (the union).

YOLO v2-

This improves on some of the shortcomings of the first version, namely the fact that it is not very good at detecting objects that are very near and tends to make some of the mistakes on localization.

It introduces a few newer things: Which are *anchor boxes* (pre-determined sets of boxes such that the network moves from predicting the bounding boxes to predicting the offsets from these) and the use of features that are more fine-grained so smaller objects can be predicted better.



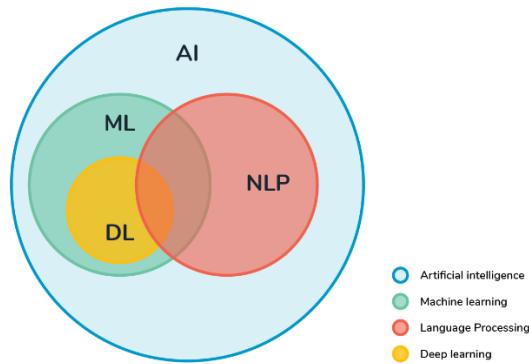
YOLO v3-

YOLOv3 came about April 2018, and it adds small improvements, including the fact that bounding boxes get predicted at the different scales. The underlying meaty part of the YOLO network, Darknet, is expanded in this version to have 53 convolutional layers



Q1. What is NLP?

Natural language processing (NLP): It is the branch of artificial intelligence that helps computers understand, interpret and manipulate human language. NLP draws from many disciplines, including computer science and computational linguistics, in its pursuit to fill the gap between human communication and computer understanding.



Q2. What are the Libraries we used for NLP?

We usually use these libraries in NLP, which are:

NLTK (Natural language Tool kit), TextBlob, CoreNLP, Polyglot,

Gensim, SpaCy, Scikit-learn

And the new one is Megatron library launched recently.

Q3. What do you understand by tokenisation?

Tokenisation is the act of breaking a sequence of strings into pieces such as words, keywords, phrases, symbols and other elements called tokens. Tokens can be individual words, phrases or even whole sentences. In the process of tokenisation, some characters like punctuation marks are discarded.

Natural Language Processing

[‘Natural’, ‘Language’, ‘Processing’]

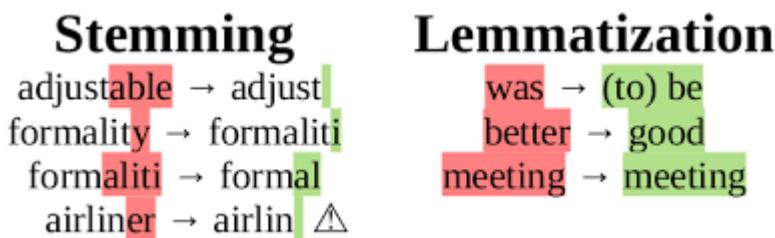
Q4. What do you understand by stemming?

Stemming: It is the process of reducing inflexions in words to their root forms such as mapping a group of words to the same stem even if stem itself is not a valid word in the Language.

	words	stemmed words
0	connect	connect
1	connected	connect
2	connection	connect
3	connections	connect
4	connects	connect

Q5. What is lemmatisation?

Lemmatisation: It is the process of the group together the different inflected forms of the word so that they can be analysed as a single item. It is quite similar to stemming, but it brings context to the words. So it links words with similar kind meaning to one word.



Q6. What is Bag-of-words model?

We need the way to represent text data for the machine learning algorithms, and the bag-of-words model helps us to achieve the task. This model is very understandable and to implement. It is the way of extracting features from the text for the use in machine learning algorithms.

In this approach, we use the tokenised words for each of observation and find out the frequency of each token.

Let's do an example to understand this concept in depth.

“It is going to rain today.”

“Today, I am not going outside.”

“I am going to watch the season premiere.”

We treat each sentence as the separate document and we make the list of all words from all the three documents excluding the punctuation. We get,

‘It’, ‘is’, ‘going’, ‘to’, ‘rain’, ‘today’ ‘I’, ‘am’, ‘not’, ‘outside’, ‘watch’, ‘the’, ‘season’, ‘premiere.’

The next step is the create vectors. Vectors convert text that can be used by the machine learning algorithm.

We take the first document — “It is going to rain today”, and we check the frequency of words from the ten unique words.

“It” = 1

“is” = 1

“going” = 1

“to” = 1

“rain” = 1

“today” = 1

“I” = 0

“am” = 0

“not” = 0

“outside” = 0

Rest of the documents will be:

“It is going to rain today” = [1, 1, 1, 1, 1, 1, 0, 0, 0, 0]

“Today I am not going outside” = [0, 0, 1, 0, 0, 1, 1, 1, 1, 1]

“I am going to watch the season premiere” = [0, 0, 1, 1, 0, 0, 1, 1, 0, 0]

In this approach, each word (a token) is called a “gram”. Creating the vocabulary of two-word pairs is called a bigram model.

The process of converting the NLP text into numbers is called **vectorisation** in ML. There are different ways to convert text into the vectors :

- *Counting the number of times that each word appears in the document.*
- *I am calculating the frequency that each word appears in a document out of all the words in the document.*

Q7.What do you understand by TF-IDF?

TF-IDF: It stands for the term of frequency-inverse document frequency.

TF-IDF weight: It is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus.

- **Term Frequency (TF):** is a scoring of the frequency of the word in the current document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. The term frequency is often divided by the document length to normalise.

$$TF(t) = \frac{\text{Number of times term } t \text{ appears in a document}}{\text{Total number of terms in the document}}$$

- **Inverse Document Frequency (IDF):** It is a scoring of how rare the word is across the documents. It is a measure of how rare a term is, Rarer the term, and more is the IDF score.

$$IDF(t) = \log_e\left(\frac{\text{Total number of documents}}{\text{Number of documents with term } t \text{ in it}}\right)$$

Thus,

$$TF - IDF \text{ score} = TF * IDF$$

Q8. What is Word2vec?

Word2Vec is a shallow, two-layer neural network which is trained to reconstruct linguistic contexts of words. It takes as its input a large corpus of words and produces a vector space, typically of several of hundred dimensions, with each of unique word in the corpus being assigned to the corresponding vector in space.

Word vectors are positioned in a vector space such that words which share common contexts in the corpus are located close to one another in the space.

Word2Vec is a particularly computationally-efficient predictive model for learning word embeddings from raw text.

Word2Vec is a group of models which helps derive relations between a word and its contextual words. Let's look at two important models inside Word2Vec: Skip-grams and CBOW.

Skip-grams

Source Text	Training Samples
The quick brown fox jumps over the lazy dog. ➔	(the, quick) (the, brown)
The quick brown fox jumps over the lazy dog. ➔	(quick, the) (quick, brown) (quick, fox)
The quick brown fox jumps over the lazy dog. ➔	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown fox jumps over the lazy dog. ➔	(fox, quick) (fox, brown) (fox, jumps) (fox, over)

In Skip-gram model, we take a centre word and a window of context (neighbour) words, and we try to predict the context of words out to some window size for each centre word. So, our model is going to define a probability distribution, i.e. probability of a word appearing in the context given a centre word and we are going to choose our vector representations to maximise the probability.

Continuous Bag-of-Words (CBOW)

CBOW predicts target words (e.g. ‘mat’) from the surrounding context words (‘the cat sits on the’).

Statistically, it affects that CBOW smoothes over a lot of distributional information (by treating an entire context as one observation). For the most part, this turns out to be a useful thing for smaller datasets.

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

The resulting counts matrix will then be:

$$X = \begin{matrix} & \begin{matrix} I & like & enjoy & deep & learning & NLP & flying & . \end{matrix} \\ \begin{matrix} I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{matrix} & \left[\begin{matrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{matrix} \right] \end{matrix}$$

This was about converting words into vectors. But where does the “learning” happen? Essentially, we begin with small random initialisation of word vectors. Our predictive model learns the vectors by minimising the loss function. In Word2vec, this happens with feed-forward neural networks and optimisation techniques such as Stochastic gradient descent. There are also count-based models which make the co-occurrence count matrix of the words in our corpus; we have a very large matrix with each row for the “words” and columns for the “context”. The number of “contexts” is, of course very large, since it is very essentially combinatorial in size. To overcome this issue, we apply SVD to a matrix. This reduces the dimensions of the matrix to retain maximum pieces of information.

Q9. What is Doc2vec?

Paragraph Vector (more popularly known as *Doc2Vec*) — Distributed Memory (*PV-DM*)

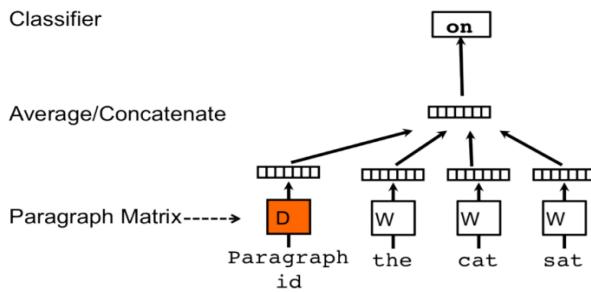
Paragraph Vector (Doc2Vec) is supposed to be an extension to Word2Vec such that *Word2Vec* learns to project words into a latent d -dimensional space whereas *Doc2Vec* aims at learning how to project a document into a latent d -dimensional space.

The basic idea behind PV-DM is inspired by Word2Vec. In CBOW model of Word2Vec, the model learns to predict a centre word based on the contexts. For example- given a sentence “The cat sat on the table”, CBOW model would learn to predict the words “sat” given the context words — the cat, on and table. Similarly,in PV-DM the main idea is: randomly sample consecutive words from the paragraph and ***predict a centre word*** from the randomly sampled set of words by taking as the ***input — the context words and the paragraph id***.

Let’s have a look at the model diagram for some more clarity. In this given model, we see Paragraph matrix, (Average/Concatenate) and classifier sections.

Paragraph matrix: It is the matrix where each column represents the vector of a paragraph.
Average/Concatenate: It means that whether the word vectors and paragraph vector are averaged or concatenated.

Classifier: In this, it takes the hidden layer vector (the one that was concatenated/averaged) as input and predicts the Centre word.



In the matrix D, It has the embeddings for “seen” paragraphs (i.e. arbitrary length documents), the same way Word2Vec models learns embeddings for words. For unseen paragraphs, the model is again run through gradient descent (5 or so iterations) to infer a document vector.

Q9. What is Time-Series forecasting?

Time series forecasting is a technique for the prediction of events through a sequence of time. The technique is used across many fields of study, from the geology to behaviour to economics. The techniques predict future events by analysing the trends of the past, on the assumption that future trends will hold similar to historical trends.

Q10. What is the difference between in Time series and regression?

Time-series:

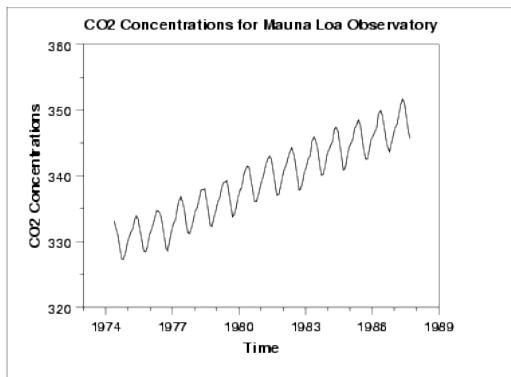
1. Whenever data is recorded at regular intervals of time.
2. Time-series forecast is Extrapolation.
3. Time-series refers to an ordered series of data.

Regression:

1. Whereas in regression, whether data is recorded at regular or irregular intervals of time, we can apply.
2. Regression is Interpolation.
3. Regression refer both ordered and unordered series of data.

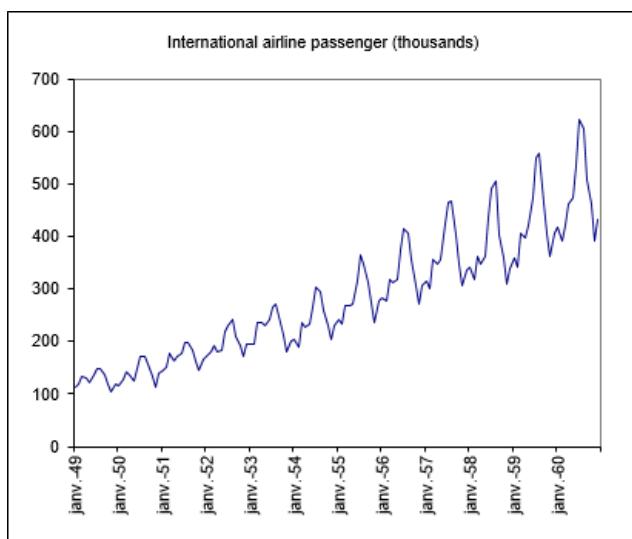
Q11. What is the difference between stationary and non-stationary data?

Stationary: A series is said to be "STRICTLY STATIONARY" if the Mean, Variance & Covariance is constant over some time or time-invariant.



Non-Stationary:

A series is said to be "STRICTLY STATIONARY" if the Mean, Variance & Covariance is not constant over some time or time-invariant.



Q12. Why you cannot take non-stationary data to solve time series Problem?

- Most models assume stationary of data. In other words, standard techniques are invalid if data is "NON-STATIONARY".
 - Autocorrelation may result due to "NON-STATIONARY".
 - Non-stationary processes are a random walk with or without a drift (a slow, steady change).
 - Deterministic trends (trends that are constant, positive or negative, independent of time for the whole life of the series).
-

Q1. What is the process to make data stationary from non-stationary in time series?

Ans:

The two most common ways to make a non-stationary time series stationary are:

- Differencing
- Transforming

Let us look at some details for each of them:

Differencing:

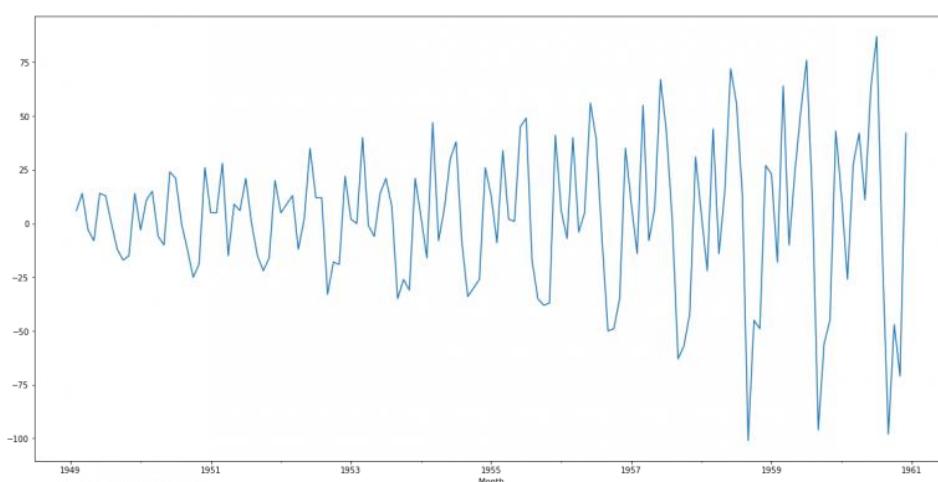
To make your series stationary, you take a difference between the data points. So let us say, your original time series was:

$X_1, X_2, X_3, \dots, X_n$

Your series with a difference of degree 1 becomes:

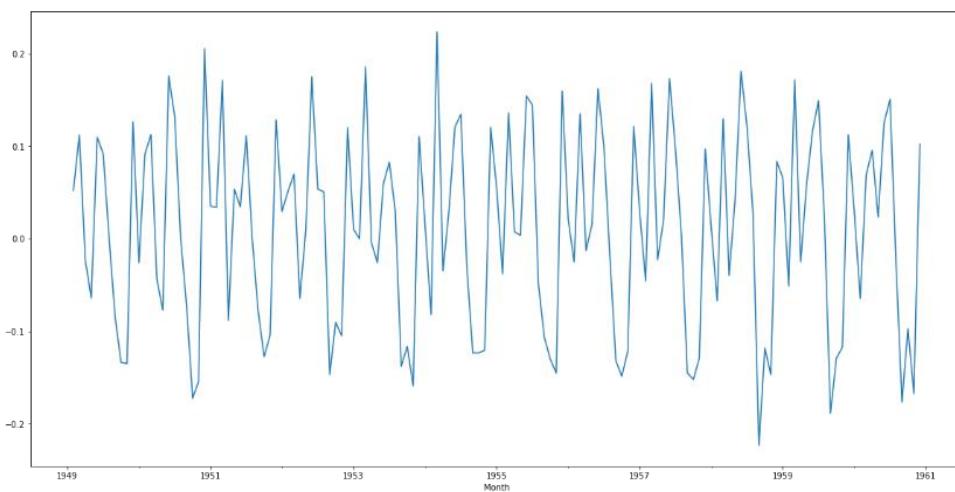
$(X_2 - X_1, X_3 - X_2, X_4 - X_3, \dots, X_n - X_{n-1})$

Once, you make the difference, plot the series and see if there is any improvement in the ACF curve. If not, you can try a second or even a third-order differencing. Remember, the more you difference, the more complicated your analysis is becoming.



Transforming:

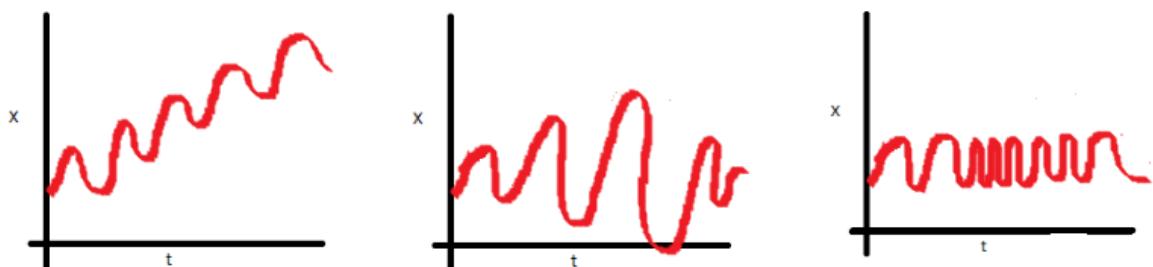
If we cannot make a time series stationary, you can try out transforming the variables. Log transform is probably the most commonly used transformation if we see the diverging time series. However, it is suggested that you use transformation only in case differencing is not working.



Q2. What is the process to check stationary data ?

Ans:

Stationary series: It is one in which the properties – mean, variance and covariance, do not vary with time.

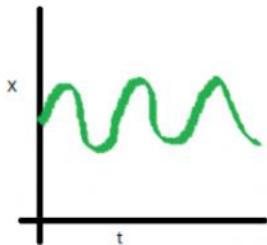


Let us get an idea with these three plots:

- In the first plot, we can see that the mean varies (increases) with time, which results in an upward trend. This is the non-stationary series.
For the series classification as stationary, it should not exhibit the trend.
- Moving on to the second plot, we do not see a trend in the series, but the variance of the series is a function of time. As mentioned previously, a stationary series must have a constant variance.

- If we look at the third plot, the spread becomes closer, as the time increases, which implies that covariance is a function of time.

These three plots refer to the non-stationary time series. Now give your attention to fourth:



In this case, Mean, Variance and Covariance are constant with time. This is how a stationary time series looks like.

Most of the statistical models require the series to be stationary to make an effective and precise prediction.

The various process you can use to find out your data is stationary or not by the following terms:

1. Visual Test
2. Statistical Test
3. ADF(Augmented Dickey-Fuller) Test
4. KPSS(Kwiatkowski-Phillips-Schmidt-Shin) Test

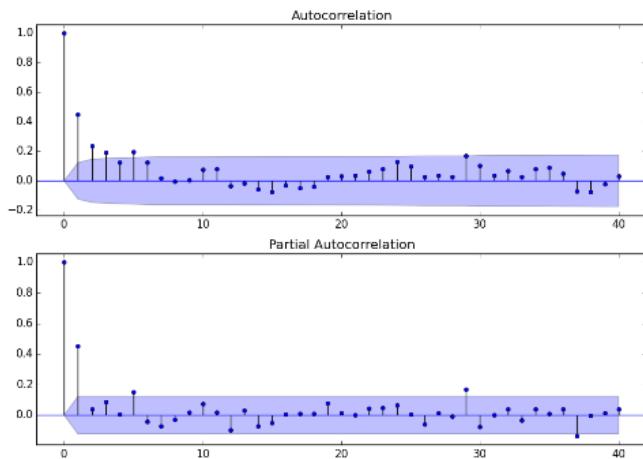
Q3. What are ACF and PACF?.

Ans:

ACF is a (complete) auto-correlation function which gives us the values of the auto-correlation of any series with lagged values. We plot these values along with a confidence band. We have an ACF plot. In simple terms, it describes how well the present value of the series is related to its past values. A time series can have components like the trend, seasonality, cyclic and residual. ACF considers all the components while finding correlations; hence, it's a 'complete auto-correlation plot'.

PACF is a partial autocorrelation function. Instead of finding correlations of present with lags like ACF, it finds the correlations of the residuals with the next lag value thus 'partial' and not 'complete' as we remove already found variations before we find next correlation. So if there are any hidden pieces of information in the residual which can be modelled by next lag, we might get a good correlation, and we'll keep that next lag as a feature while modelling. Remember, while

modelling we don't want to keep too many correlated features, as that it can create multicollinearity issues. Hence we need to retain only relevant features.

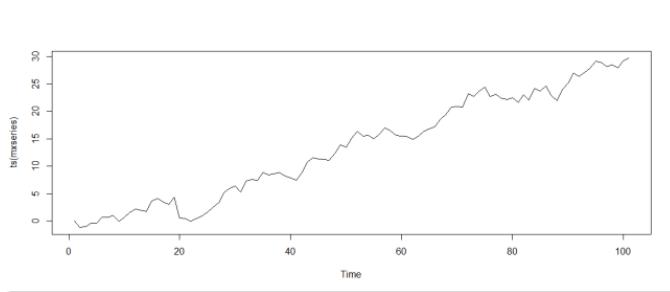


Q4. What do you understand by the trend of data?

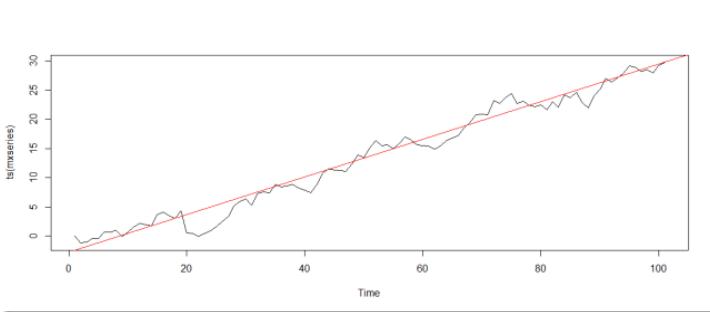
Ans:

A general systematic linear or (most often) nonlinear component that changes over time and does not repeat.

There are different approaches to understanding trend. A positive trend means it is likely that growth continues. Let's illustrate this with a simple example:



Hmm, this looks like there is a trend. To build up confidence, let's add a linear regression for this graph:



Great, now it's clear there's a trend in the graph by adding Linear Regression.

Q5. What is the Augmented Dickey-Fuller Test?

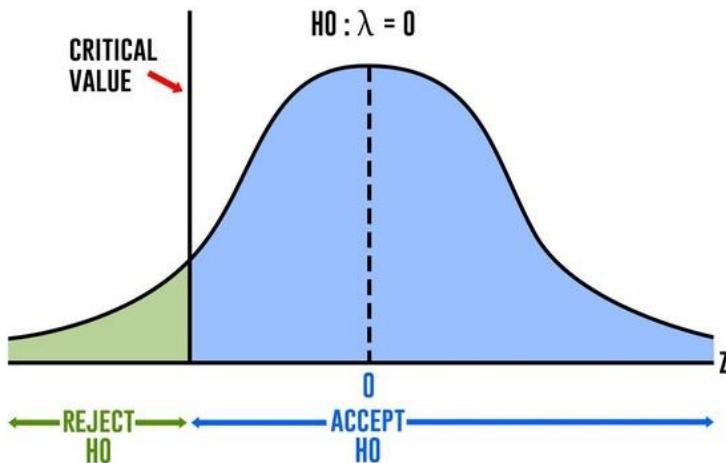
Ans:

The Dickey-Fuller test: It is one of the most popular statistical tests. It is used to determine the presence of unit root in a series, and hence help us to understand if the series is stationary or not. The null and alternate hypothesis for this test is:

Null Hypothesis: The series has a unit root (value of $\alpha = 1$)

Alternate Hypothesis: The series has no unit root.

If we fail to reject the null hypothesis, we can say that the series is non-stationary. This means that the series can be linear or difference stationary.



Q6. What is AIC and BIC into time series?

Ans:

Akaike's information criterion (AIC) compares the quality of a set of statistical models to each other. For example, you might be interested in what variables contribute to low socioeconomic status and how the variables contribute to that status. Let's say you create several regression models for various factors like education, family size, or disability status; The AIC will take each model and rank them from best to worst. The “best” model will be the one that neither under-fits nor over-fits.

- AIC
- K = number of estimated parameters in the model
- L = Maximized likelihood function for the estimated model

$$AIC = 2k - 2 \ln(L)$$

The Bayesian Information Criterion (BIC) can be defined as:

$$k \log(n) - 2 \log(L(\hat{\theta})).$$

Here n is the sample size.

K is the number of parameters which your model estimates.

Θ is the set of all parameter.

$L(\hat{\theta})$ represents the likelihood of the model tested, when evaluated at maximum likelihood values of θ .

Q7. What are the components of the Time -Series?

Ans:

Time series analysis: It provides a body of techniques to understand a dataset better. The most useful one is the decomposition of the time series into four constituent parts-

1. Level- The baseline value for the series if it were a straight line.
2. Trend - The optional and linear, increasing or decreasing behaviour of series over time.
3. Seasonality - Optional repeated patterns /cycles of behaviour over time.
4. Noise - The optional variability in the observations that cannot be explained by the model.

Q8. What is Time Series Analysis?

Ans:

Time series analysis: It involves developing models that best capture or describe an observed time series to understand the underlying cause. This study seeks the “why” behind the time-series datasets. This involves making assumptions about the form of data and decomposing time-series into the constitution component.

Quality of descriptive model is determined by how well it describes all available data and the interpretation it provides to inform the problem domain better.

Q9. Give some examples of the Time-Series forecast?

Ans:

There is almost an endless supply of the time series forecasting problems. Below are ten examples from a range of industries to make the notions of time series analysis and forecasting more concrete.

1. Forecasting the corn yield in tons by the state each year.
2. Forecasting whether an EEG trace in seconds indicates a patient is having a seizure or not.
3. Forecasting the closing price of stocks every day.
4. Forecasting the birth rates at all hospitals in the city every year.
5. Forecasting product sales in the units sold each day for the store.
6. Forecasting the number of passengers through the train station each day.
7. Forecasting unemployment for a state each quarter.
8. Forecasting the utilisation demand on the server every hour.
9. Forecasting the size of the rabbit populations in the state each breeding season.
10. Forecasting the average price of gasoline in a city each day.

Q10. What are the techniques of Forecasting?

Ans:

There are so many statistical techniques available for time series forecast however we have found a few effective ones which are listed below:

- Simple Moving Average (SMA)
- Exponential Smoothing (SES)
- Autoregressive Integration Moving Average (ARIMA)

Q11. What is the Moving Average?

Ans:

The moving average model is probably the most naive approach to time series modelling. This model states that the next observation is the mean of all past observations.

Although simple, this model might be surprisingly good, and it represents a good starting point.

Otherwise, the moving average can be used to identify interesting trends in the data. We can define a window to apply the moving average model to smooth the time series and highlight different trends.



Example of a moving average on a 24h window

In the plot above, we applied the moving average model to a 24h window. The green line smoothed the time series, and we can see that there are two peaks in the 24h period.

The longer the window, the smoother the trend will be.

Below is an example of moving average on a smaller window.



Example of a moving average on a 12h window

Q12. What is Exponential smoothing?

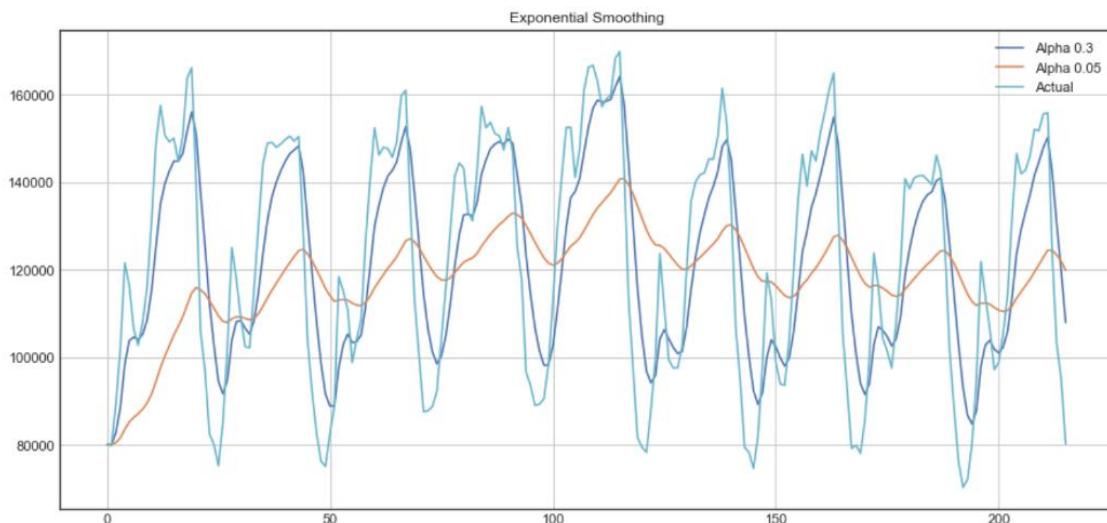
Ans:

Exponential smoothing uses similar logic to moving average, but this time, different decreasing weight is assigned to each observation. We can also say, less importance is given to the observations as we move further from the present.

Mathematically, exponential smoothing is expressed as:

$$y = \alpha x_t + (1 - \alpha)y_{t-1}, t > 0$$

Here, alpha is the smoothing factor which takes values between 0 to 1. It determines how fast the weight will decrease for the previous observations.



From the above plot, the dark blue line represents the exponential smoothing of the time series using a smoothing factor of 0.3, and the orange line uses a smoothing factor of 0.05. As we can see, the smaller the smoothing factor, the smoother the time series will be. Because as smoothing factor approaches 0, we approach to the moving average model

Q1. What is Tensorflow?

Ans:

TensorFlow: TensorFlow is an open-source software library released in 2015 by Google to make it easier for the developers to design, build, and train deep learning models. TensorFlow is originated as an internal library that the Google developers used to build the models in house, and we expect additional functionality to be added in the open-source version as they are tested and vetted in internal flavour. Although TensorFlow is the only one of several options available to the developers and we choose to use it here because of thoughtful design and ease of use.

At a high level, TensorFlow is a Python library that allows users to express arbitrary computation as a graph of *data flows*. Nodes in this graph represent mathematical operations, whereas edges represent data that is communicated from one node to another. Data in TensorFlow are represented as tensors, which are multidimensional arrays. Although this framework for thinking about computation is valuable in many different fields, TensorFlow is primarily used for deep learning in practice and research.

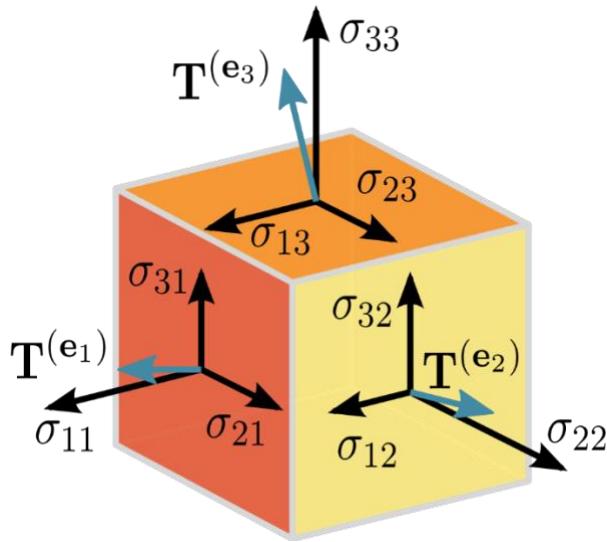


Q2. What are Tensors?

Ans:

Tensor: In mathematics, it is an algebraic object that describes the linear mapping from one set of algebraic objects to the another. Objects that the tensors may map between include, but are not limited to the vectors, scalars and recursively, even other tensors (for example, a matrix is the map between vectors and thus a tensor. Therefore the linear map between matrices is also the tensor). Tensors are inherently related to the vector spaces and their dual spaces and can take several different forms. For

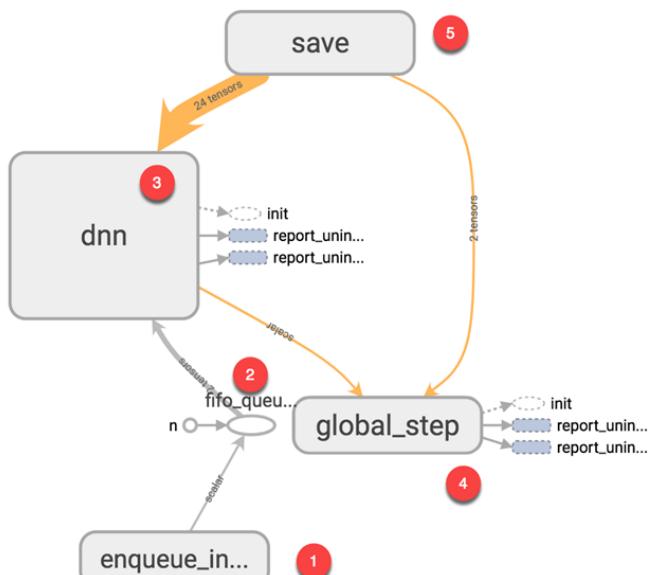
example, a scalar, a vector, a dual vector at a point, or a multi-linear map between vector spaces. Euclidean vectors and scalars are simple tensors. While tensors are defined as independent of any basis. The literature on physics, often referred by their components on a basis related to a particular coordinate system.



Q3. What is TensorBoard?

Ans:

TensorBoard, a suite of visualising tools, is an easy solution to Tensorflow offered by the creators that lets you visualise the graphs, plot quantitative metrics about the graph with additional data like images to pass through it.



This one is some example of how the TensorBoard is working.

Q4. What are the features of TensorFlow?

Ans:

- One of the main features of TensorFlow is its ability to build neural networks.
- By using these neural networks, machines can perform logical thinking and learn similar to humans.
- There are other tensors for processing, such as data loading, preprocessing, calculation, state and outputs.
- It is considered not only as deep learning but also as the library for performing tensor calculations, and it is the most excellent library when considered as the deep learning framework that can also describe basic calculation processing.
- TensorFlow describes all calculation processes by calculation graph, no matter how simple the calculation is.

Q5. What are the advantages of TensorFlow?

Ans:

- It allows Deep Learning.
- It is open-source and free.
- It is reliable (and without major bugs)
- It is backed by Google and a good community.
- It is a skill recognised by many employers.
- It is easy to implement.

Q6. List a few limitations of Tensorflow.

Ans:

- Has GPU memory conflicts with Theano if imported in the same scope.
- It has dependencies with other libraries.
- Requires prior knowledge of advanced calculus and linear algebra along with a pretty good understanding of machine learning.

Q7. What are the use cases of Tensor flow?

Ans:

Tensorflow is an important tool of deep learning, it has mainly five use cases, and they are:

- Time Series
- Image recognition
- Sound Recognition
- Video detection
- Text-based Applications

Q8. What are the very important steps of Tensorflow architecture?

Ans:

There are three main steps in the Tensorflow architecture are:

- Pre-process the Data
- Build a Model
- Train and estimate the model

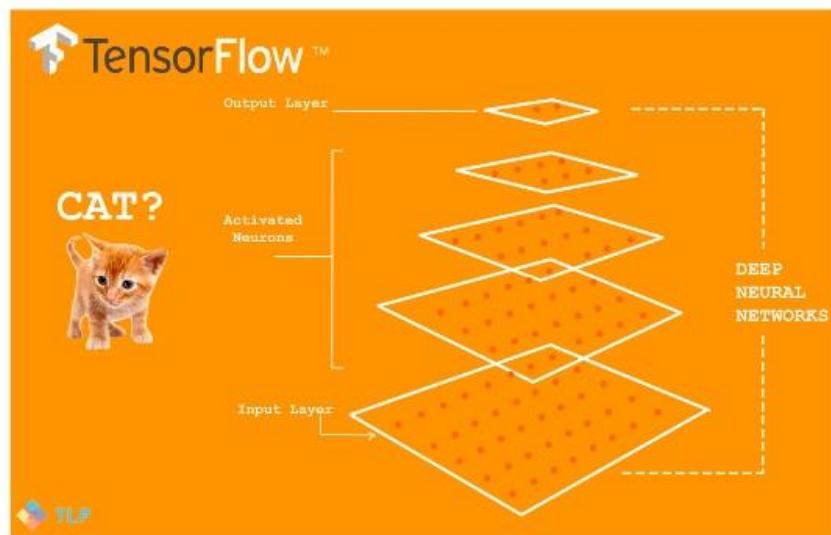
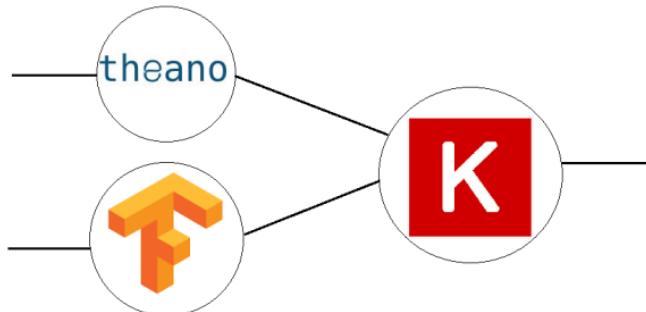


Image Recognition
Classification using Softmax Regressions and Convolutional Neural Networks

Q9. What is Keras?

Ans:

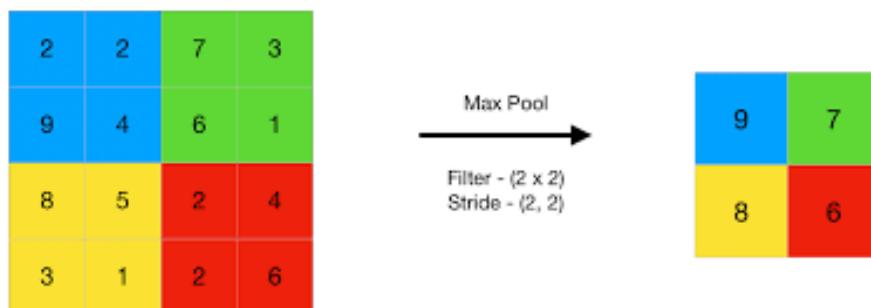
Keras: It is an Open Source Neural Network library written in Python that runs on the top of Theano or Tensorflow. It is designed to be the modular, fast and easy to use. It was developed by François Chollet, a Google engineer.



Q10. What is a pooling layer?

Ans:

Pooling layer: It is generally used in reducing the spatial dimensions and not depth, on a convolutional neural network model.



Q11. What is the difference between CNN and RNN?

Ans:

CNN (Convolutional Neural Network)

- Best suited for spatial data like images
- CNN is powerful compared to RNN
- This network takes a fixed type of inputs and outputs
- These are the ideal for video and image processing

RNN (Recurrent Neural Network)

- Best suited for sequential data
- RNN supports less feature set than CNN.
- This network can manage the arbitrary input and output lengths.
- It is ideal for text and speech analysis.

Q12. What are the benefits of Tensorflow over other libraries?

Ans:

The following benefits are:

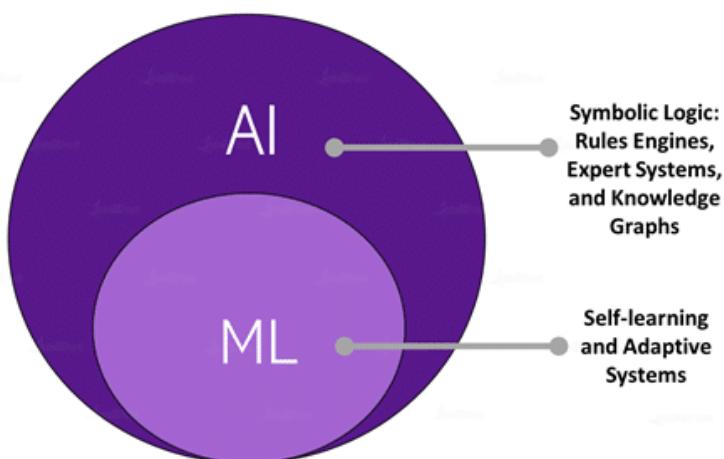
- Scalability
 - Visualisation of Data
 - Debugging facility
 - Pipelining
-

Q1: How would you define Machine Learning?

Ans:

Machine learning: It is an application of artificial intelligence (AI) that provides systems the ability to learn automatically and to improve from experiences without being programmed. It focuses on the development of computer applications that can access the data and used it to learn for themselves.

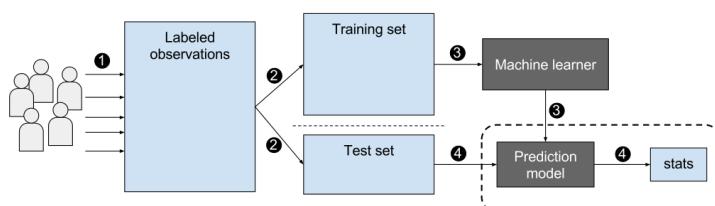
The process of learning starts with the observations or data, such as examples, direct experience, or instruction, to look for the patterns in data and to make better decisions in the future based on examples that we provide. The primary aim is to allow the computers to learn automatically without human intervention or assistance and adjust actions accordingly.



Q2. What is a labeled training set?

Ans:

Machine learning is derived from the availability of the labeled data in the form of a **training set** and **test set** that is used by the learning algorithm. The separation of data into the training portion and a test portion is the way the algorithm learns. We split up the data containing known response variable values into two pieces. The training set is used to train the algorithm, and then you use the trained model on the test set to predict the variable response values that are already known. The final step is to compare with the predicted responses against actual (observed) responses to see how close they are. The difference is the test error metric. Depending on the test error, you can go back to refine the model and repeat the process until you're satisfied with the accuracy.



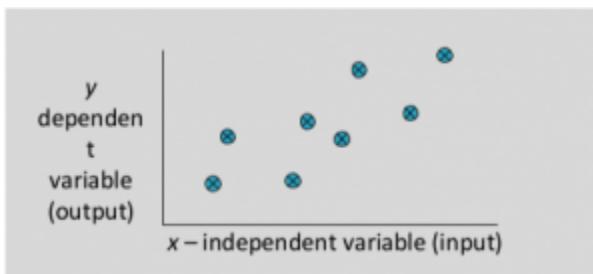
Q3. What are the two common supervised tasks?

Ans:

The two common supervised tasks are regression and classification.

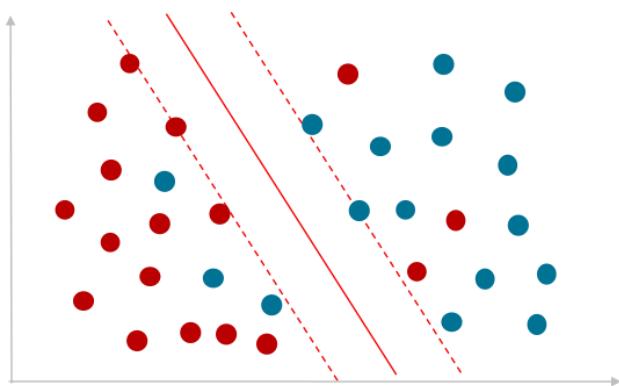
Regression-

The regression problem is when the output variable is the real or continuous value, such as “salary” or “weight.” Many different models can be used, and the simplest is linear regression. It tries to fit the data with the best hyper-plane, which goes through the points.



Classification

It is the type of supervised learning. It specifies the class to which the data elements belong to and is best used when the output has finite and discrete values. It predicts a class for an input variable, as well.



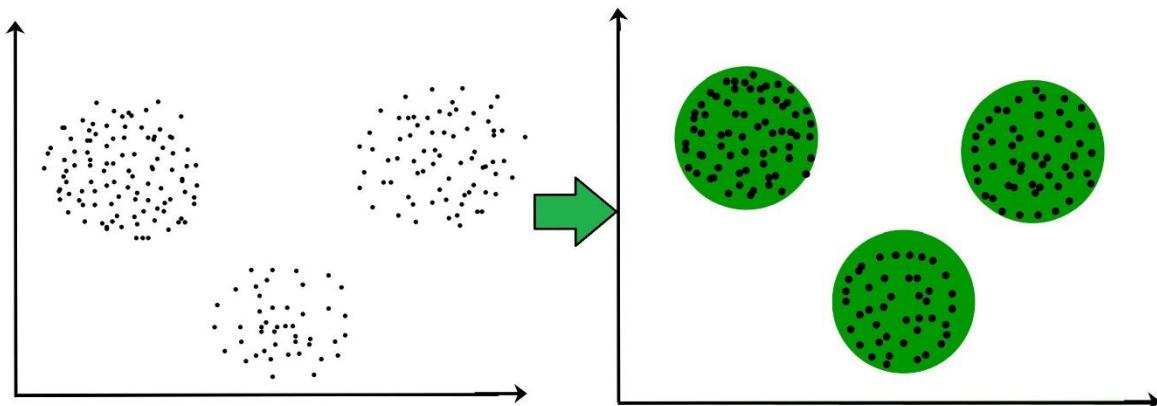
Q4. Can you name four common unsupervised tasks?

Ans:

The common unsupervised tasks include clustering, visualization, dimensionality reduction, and association rule learning.

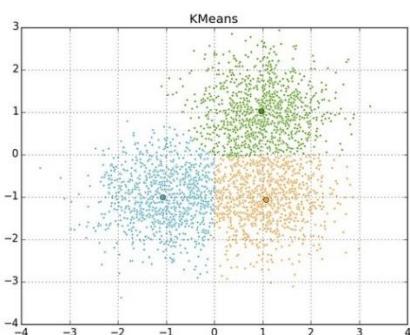
Clustering

It is a Machine Learning technique that involves the grouping of the data points. Given a set of data points, and we can use a clustering algorithm to classify each data point into the specific group. In theory, data points that lie in the same group should have similar properties and/ or features, and data points in the different groups should have high dissimilar properties and/or features. Clustering is the method of unsupervised learning and is a common technique for statistical data analysis used in many fields.



Visualization

Data visualization is the technique that uses an array of static and interactive visuals within the specific context to help people to understand and make sense of the large amounts of data. The data is often displayed in the story format that visualizes patterns, trends, and correlations that may go otherwise unnoticed. It is regularly used as an avenue to monetize data as the product. An example of using monetization and data visualization is Uber. The app combines visualization with real-time data so that customers can request a ride.



Q5. What type of Machine Learning algorithm we use to allow a robot to walk in various unknown terrains?

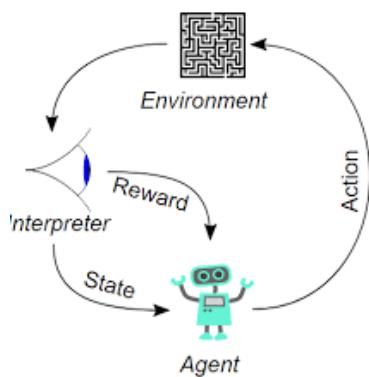
Ans:

Reinforcement Learning is likely to perform the best if we want a robot to learn how to walk in the various unknown terrains since this is typically the type of problem that the reinforcement learning

tackles. It may be possible to express the problem as a supervised or semisupervised learning problem, but it would be less natural.

Reinforcement Learning-

It's about to take suitable actions to maximize rewards in a particular situation. It is employed by the various software and machines to find out the best possible behavior/path it should take in specific situations. Reinforcement learning is different from the supervised learning in a way that in supervised learning, training data has answer key with it so that the model is trained with the correct answer itself, but in reinforcement learning, there is no answer, and the reinforcement agent decides what to do to perform the given task. In the absence of the training dataset, it is bound to learn from its experience.



Q6. What type of algorithm would we use to segment your customers into multiple groups?

Ans:

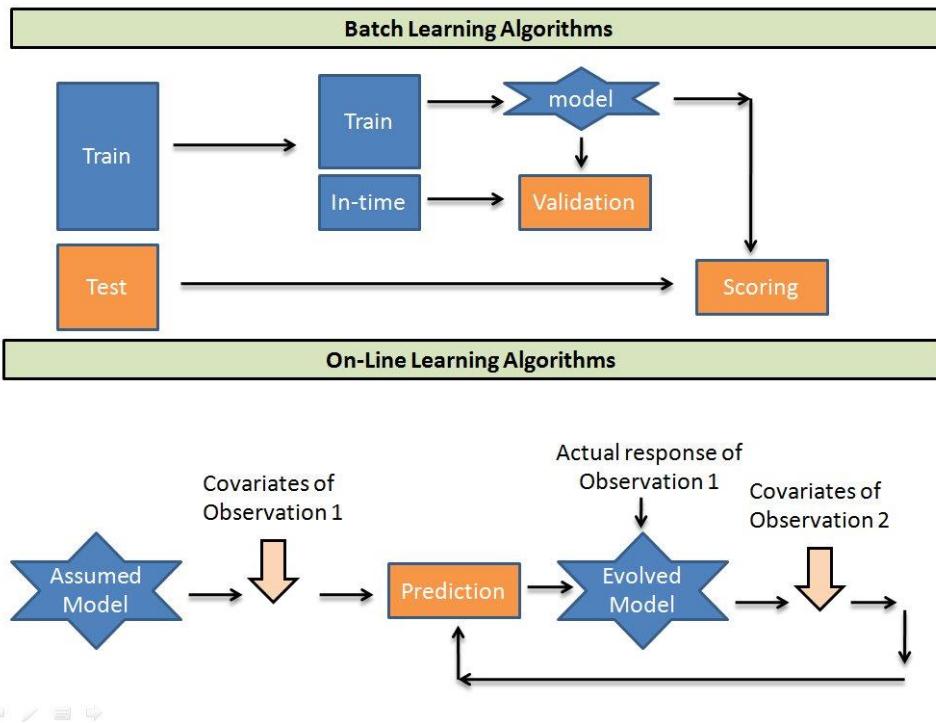
If we don't know how to define the groups, then we can use the clustering algorithm (unsupervised learning) to segment our customers into clusters of similar customers. However, if we know what groups we would like to have, then we can feed many examples of each group to a classification algorithm (supervised learning), and it will classify all your customers into these groups.

Q7: What is an online machine learning?

Ans:

Online machine learning: It is a method of machine learning in which data becomes available in sequential order and to update our best predictor for the future data at each step, as opposed to batch learning techniques that generate the best predictor by learning on entire training data set at once. Online learning is a common technique and used in the areas of machine learning where it is computationally infeasible to train over the datasets, requiring the need for Out-of-core algorithms. It is also used in situations where the algorithm must adapt to new patterns in the data dynamically or when the data itself is generated as the function of time, for example, stock

price prediction. Online learning algorithms might be prone to catastrophic interference and problem that can be addressed by the incremental learning approaches.



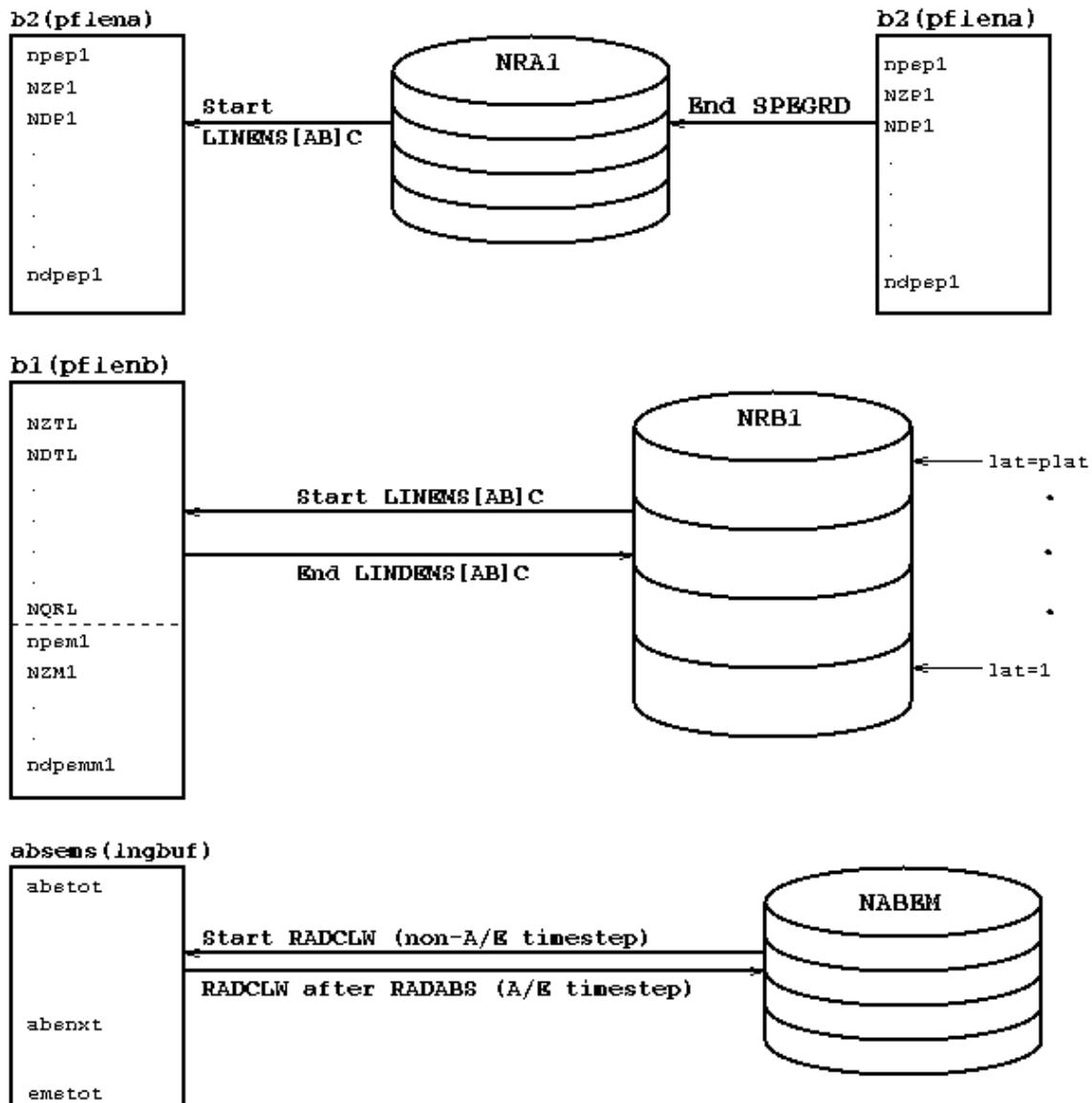
Q8: What is out-of-core learning?

Ans:

Out-of-core: It refers to the processing data that is too large to fit into the computer's main memory. Typically, when the dataset fits neatly into the computer's main memory, randomly accessing sections of data has a (relatively) small performance penalty.

When data must be stored in a medium like a large spinning hard drive or an external computer network, it becomes very expensive to seek an arbitrary section of data randomly or to process the same data multiple times. In such a case, an out-of-core algorithm will try to access all the relevant data in a sequence.

However, modern computers have deep memory hierarchy, and replacing random access with the sequential access can increase the performance even on datasets that fit within memory.



Q9. What is the Model Parameter?

Ans:

Model parameter: It is a configuration variable that is internal to a model and whose value can be predicted from the data.

- While making predictions, the model parameter is needed.
- The values define the skill of a model on problems.
- It is estimated or learned from data.
- It is often not set manually by the practitioner.
- It is often saved as part of the learned model.

Parameters are key to machine learning algorithms. They are part of the model that is learned from historical training data.

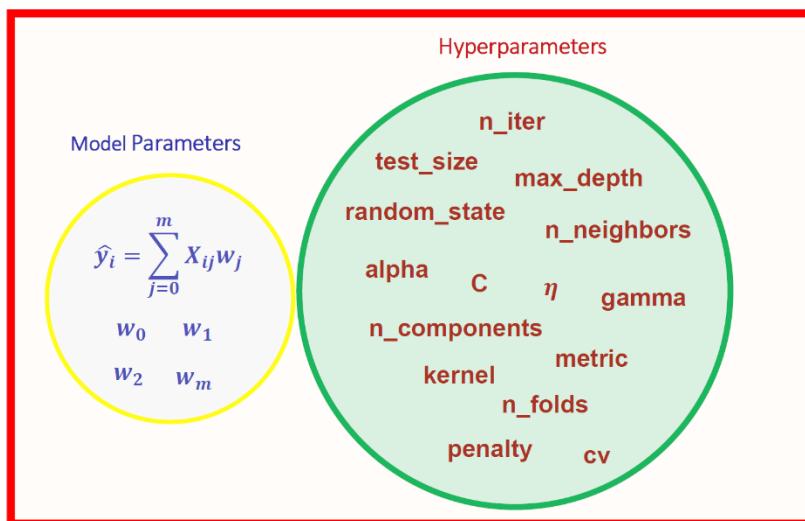
Q11: What is Model Hyperparameter?

Ans:

Model hyperparameter: It is a configuration that is external to a model and whose values cannot be estimated from the data.

- It is often used in processes to help estimate model parameters.
- The practitioner often specifies them.
- It can often be set using heuristics.
- It is tuned for the given predictive modeling problems.

We cannot know the best value for the model hyperparameter on the given problem. We may use the rules of thumb, copy values used on other problems, or search for the best value by trial and error.



Q12. What is cross-validation?

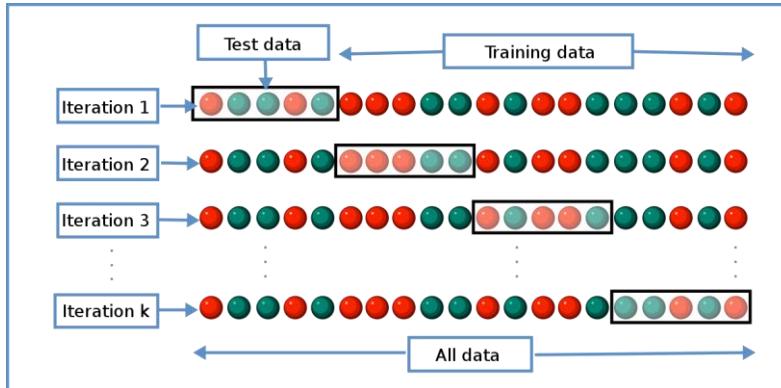
Ans:

Cross-validation: It is a technique for evaluating Machine Learning models by training several Machine Learning models on subsets of available input data and evaluating them on the complementary subset of data. Use cross-validation to detect overfitting, i.e., failing to generalize a pattern.

There are three steps involved in cross-validation are as follows :

- Reserve some portion of the sample dataset.

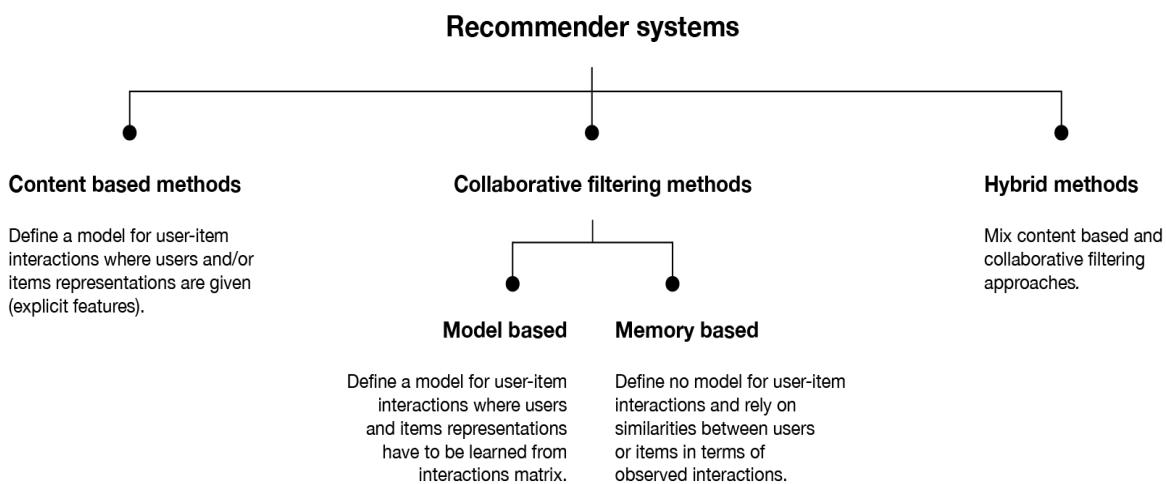
- Using the rest dataset and train models.
- Test the model using a reserve portion of the data-set.



Q1. What is a Recommender System?

Answer:

A recommender system is today widely deployed in multiple fields like movie recommendations, music preferences, social tags, research articles, search queries and so on. The recommender systems work as per collaborative and content-based filtering or by deploying a personality-based approach. This type of system works based on a person's past behavior in order to build a model for the future. This will predict the future product buying, movie viewing or book reading by people. It also creates a filtering approach using the discrete characteristics of items while recommending additional items.



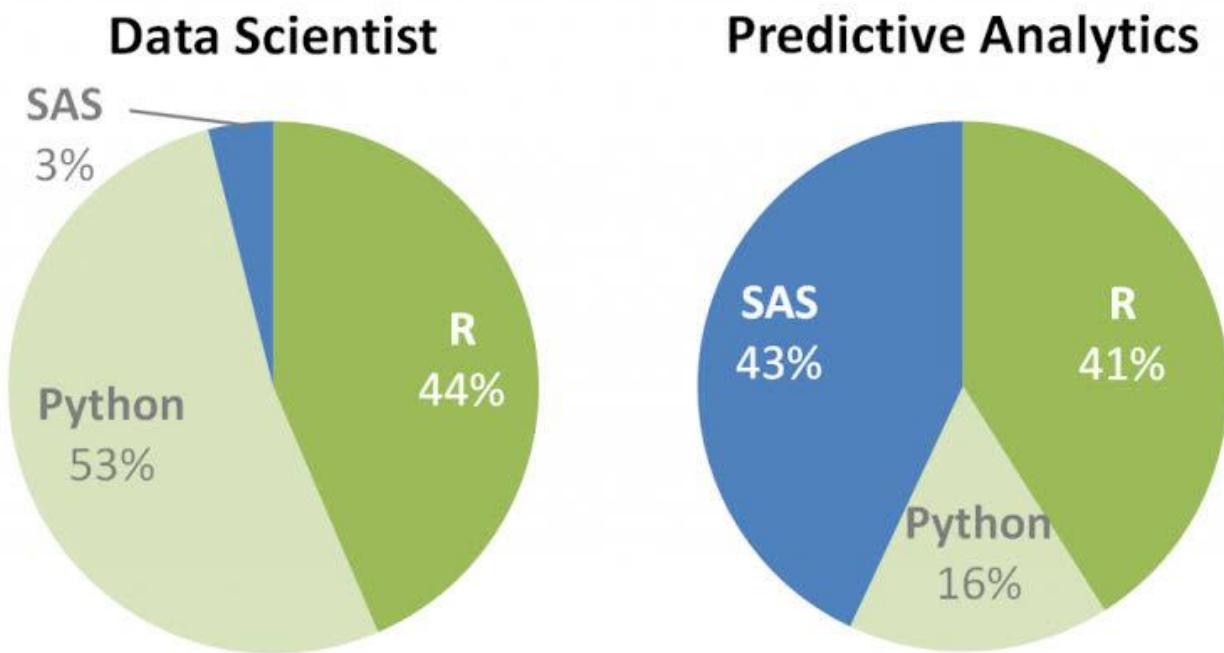
Q2. Compare SAS, R and Python programming?

Answer:

SAS: it is one of the most widely used analytics tools used by some of the biggest companies on earth. It has some of the best statistical functions, graphical user interface, but can come with a price tag and hence it cannot be readily adopted by smaller enterprises

R: The best part about R is that it is an Open Source tool and hence used generously by academia and the research community. It is a robust tool for statistical computation, graphical representation and reporting. Due to its open source nature it is always being updated with the latest features and then readily available to everybody.

Python: Python is a powerful open source programming language that is easy to learn, works well with most other tools and technologies. The best part about Python is that it has innumerable libraries and community created modules making it very robust. It has functions for statistical operation, model building and more.



Q3. Why is important in data analysis?

Answer:

With data coming in from multiple sources it is important to ensure that data is good enough for analysis. This is where data cleansing becomes extremely vital. Data cleansing extensively deals with the process of detecting and correcting of data records, ensuring that data is complete and accurate and the components of data that are irrelevant are deleted or modified as per the needs. This process can be deployed in concurrence with data wrangling or batch processing.

Once the data is cleaned it confirms with the rules of the data sets in the system. Data cleansing is an essential part of the data science because the data can be prone to error due to human negligence, corruption during transmission or storage among other things. Data cleansing takes a huge chunk of time and effort of a Data Scientist because of the multiple sources from which data emanates and the speed at which it comes.



Q4. What are the various aspects of a Machine Learning process?

Answer:

Here we will discuss the components involved in solving a problem using machine learning.

Domain knowledge

This is the first step wherein we need to understand how to extract the various features from the data and learn more about the data that we are dealing with. It has got more to do with the type of domain that we are dealing with and familiarizing the system to learn more about it.

Feature Selection

This step has got more to do with the feature that we are selecting from the set of features that we have. Sometimes it happens that there are a lot of features and we have to make an intelligent decision regarding the type of feature that we want to select to go ahead with our machine learning endeavor.

Algorithm

This is a vital step since the algorithms that we choose will have a very major impact on the entire process of machine learning. You can choose between the linear and nonlinear algorithm. Some of the algorithms used are Support Vector Machines, Decision Trees, Naïve Bayes, K-Means Clustering, etc.

Training

This is the most important part of the machine learning technique and this is where it differs from the traditional programming. The training is done based on the data that we have and providing more real world experiences. With each consequent training step the machine gets better and smarter and able to take improved decisions.

Evaluation

In this step we actually evaluate the decisions taken by the machine in order to decide whether it is up to the mark or not. There are various metrics that are involved in this process and we have to closed deploy each of these to decide on the efficacy of the whole machine learning endeavor.

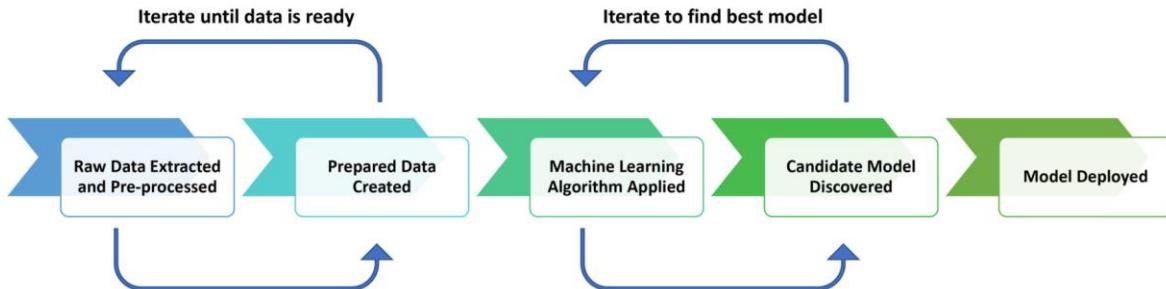
Optimization

This process involves improving the performance of the machine learning process using various optimization techniques. Optimization of machine learning is one of the most vital components wherein the performance of the algorithm is vastly improved. The best part of optimization techniques is that machine learning is not just a consumer of optimization techniques but it also provides new ideas for optimization too.

Testing

Here various tests are carried out and some these are unseen set of test cases. The data is partitioned into test and training set. There are various testing techniques like cross-validation in order to deal with multiple situations.

The Machine Learning Process

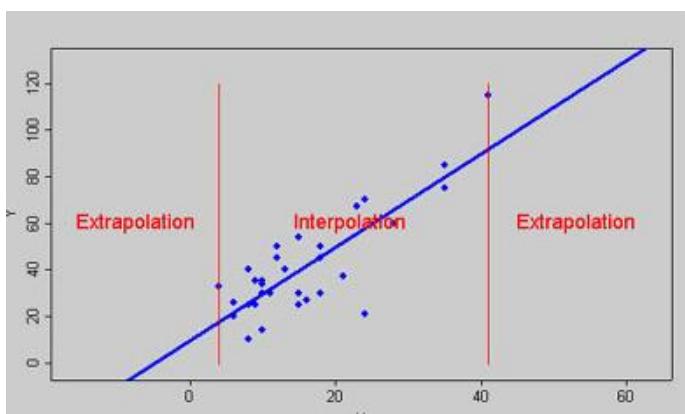


Q4. What is Interpolation and Extrapolation?

Answer:

The terms of interpolation and extrapolation are extremely important in any statistical analysis. Extrapolation is the determination or estimation using a known set of values or facts by extending it and taking it to an area or region that is unknown. It is the technique of inferring something using data that is available.

Interpolation on the other hand is the method of determining a certain value which falls between a certain set of values or the sequence of values. This is especially useful when you have data at the two extremities of a certain region but you don't have enough data points at the specific point. This is when you deploy interpolation to determine the value that you need.

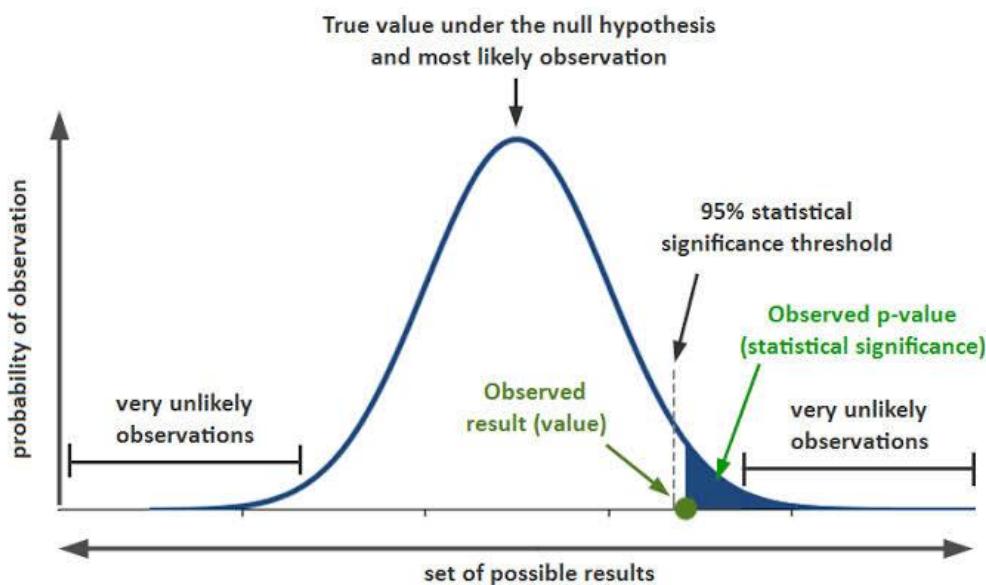


Q5. What does P-value signify about the statistical data?

Answer:

P-value is used to determine the significance of results after a hypothesis test in statistics. P-value helps the readers to draw conclusions and is always between 0 and 1.

- P- Value > 0.05 denotes weak evidence against the null hypothesis which means the null hypothesis cannot be rejected.
- P-value ≤ 0.05 denotes strong evidence against the null hypothesis which means the null hypothesis can be rejected.
- P-value=0.05 is the marginal value indicating it is possible to go either way.



Q6. During analysis, how do you treat missing values?

Answer:

The extent of the missing values is identified after identifying the variables with missing values. If any patterns are identified the analyst has to concentrate on them as it could lead to interesting and meaningful business insights. If there are no patterns identified, then the missing values can be substituted with mean or median values (imputation) or they can simply be ignored.

There are various factors to be considered when answering this question-

Understand the problem statement, understand the data and then give the answer. Assigning a default value which can be mean, minimum or maximum value. Getting into the data is important.

If it is a categorical variable, the default value is assigned. The missing value is assigned a default value.

If you have a distribution of data coming, for normal distribution give the mean value.

Should we even treat missing values is another important point to consider? If 80% of the values for a variable are missing then you can answer that you would be dropping the variable instead of treating the missing values.

Q7. Explain the difference between a Test Set and a Validation Set?

Answer:

Validation set can be considered as a part of the training set as it is used for parameter selection and to avoid Overfitting of the model being built. On the other hand, test set is used for testing or evaluating the performance of a trained machine learning model.

In simple terms ,the differences can be summarized as-

Training Set is to fit the parameters i.e. weights.

Test Set is to assess the performance of the model i.e. evaluating the predictive power and generalization.

Validation set is to tune the parameters.



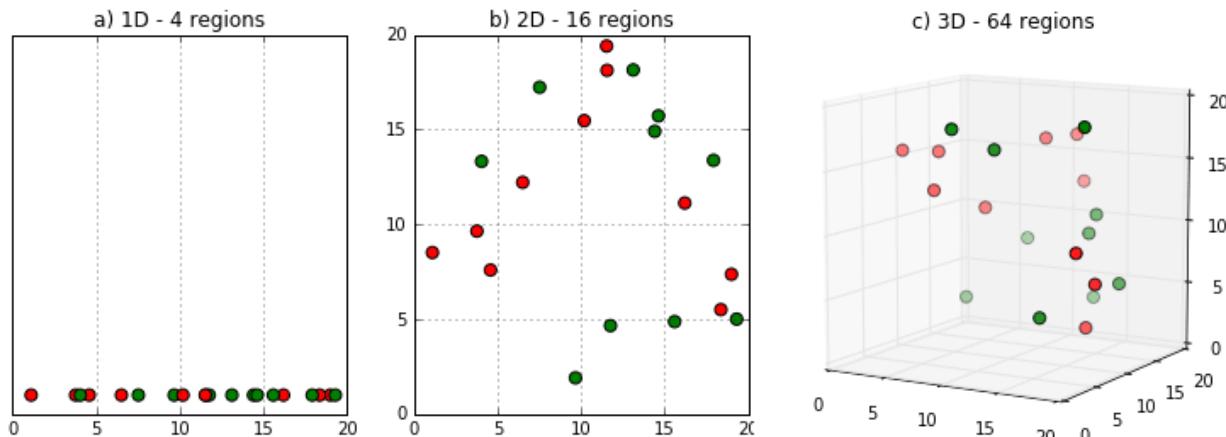
Q8. What is the curse of dimensionality? Can you list some ways to deal with it?

Answer:

The curse of dimensionality is when the training data has a high feature count, but the dataset does not have enough samples for a model to learn correctly from so many features. For example, a training dataset of 100 samples with 100 features will be very hard to learn from because the model will find random relations between the features and the target. However, if we had a dataset of 100k samples with 100 features, the model could probably learn the correct relationships between the features and the target.

There are different options to fight the curse of dimensionality:

- **Feature selection.** Instead of using all the features, we can train on a smaller subset of features.
- **Dimensionality reduction.** There are many techniques that allow to reduce the dimensionality of the features. Principal component analysis (PCA) and using autoencoders are examples of dimensionality reduction techniques.
- **L1 regularization.** Because it produces sparse parameters, L1 helps to deal with high-dimensionality input.
- **Feature engineering.** It's possible to create new features that sum up multiple existing features. For example, we can get statistics such as the mean or median.



Q9. What is data augmentation? Can you give some examples?

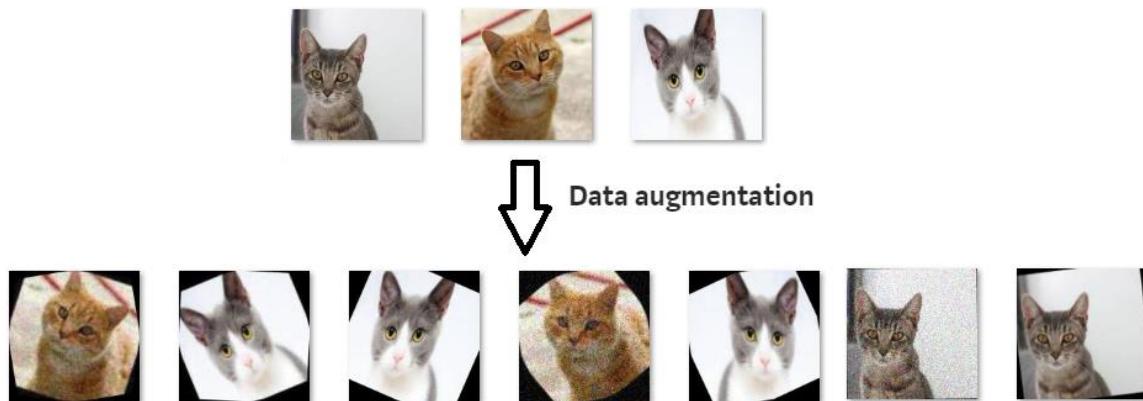
Answer:

Data augmentation is a technique for synthesizing new data by modifying existing data in such a way that the target is not changed, or it is changed in a known way.

Computer vision is one of fields where data augmentation is very useful. There are many modifications that we can do to images:

- Resize
- Horizontal or vertical flip
- Rotate
- Add noise
- Deform
- Modify colors

Each problem needs a customized data augmentation pipeline. For example, on OCR, doing flips will change the text and won't be beneficial; however, resizes and small rotations may help.



Q10. What is stratified cross-validation and when should we use it?

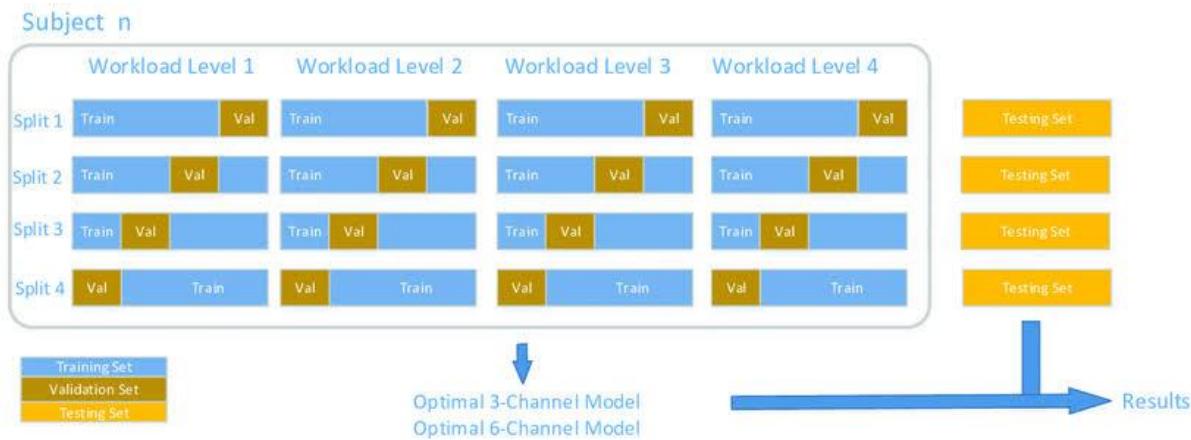
Answer:

Cross-validation is a technique for dividing data between training and validation sets. On typical cross-validation this split is done randomly. But in *stratified* cross-validation, the split preserves the ratio of the categories on both the training and validation datasets.

For example, if we have a dataset with 10% of category A and 90% of category B, and we use stratified cross-validation, we will have the same proportions in training and validation. In contrast, if we use simple cross-validation, in the worst case we may find that there are no samples of category A in the validation set.

Stratified cross-validation may be applied in the following scenarios:

- **On a dataset with multiple categories.** The smaller the dataset and the more imbalanced the categories, the more important it will be to use stratified cross-validation.
- **On a dataset with data of different distributions.** For example, in a dataset for autonomous driving, we may have images taken during the day and at night. If we do not ensure that both types are present in training and validation, we will have generalization problems.

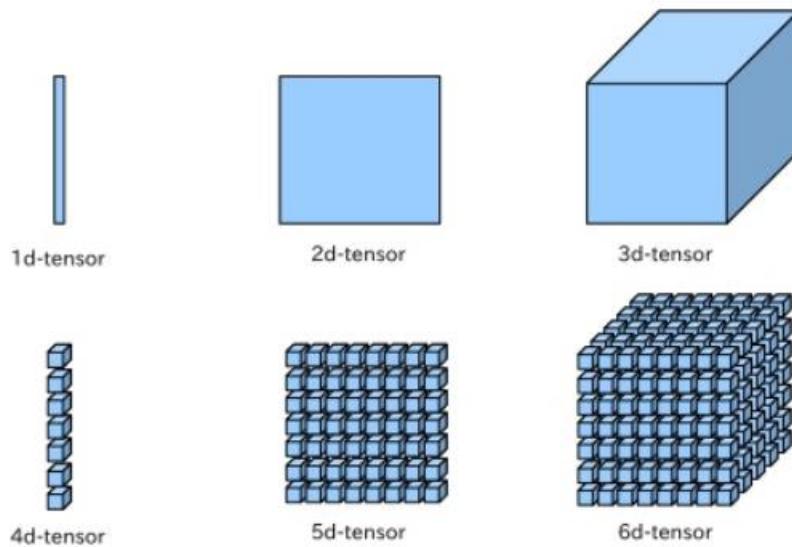


Q1. What are tensors?

Answer:

The tensors are no more than a method of presenting the data in deep learning. If put in the simple term, tensors are just multidimensional arrays that allow developers to represent the data in a layer, which means deep learning you are using contains high-level data sets where each dimension represents a different feature.

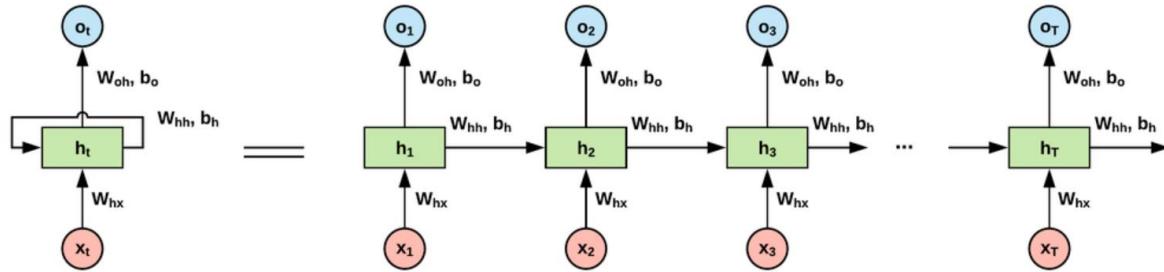
The foremost benefit of using tensors is it provides the much-needed platform-flexibility and is easy to trainable on CPU. Apart from this, tensors have the auto differentiation capabilities, advanced support system for queues, threads, and asynchronous computation. All these features also make it customizable.



Q2. Define the concept of RNN?

Answer:

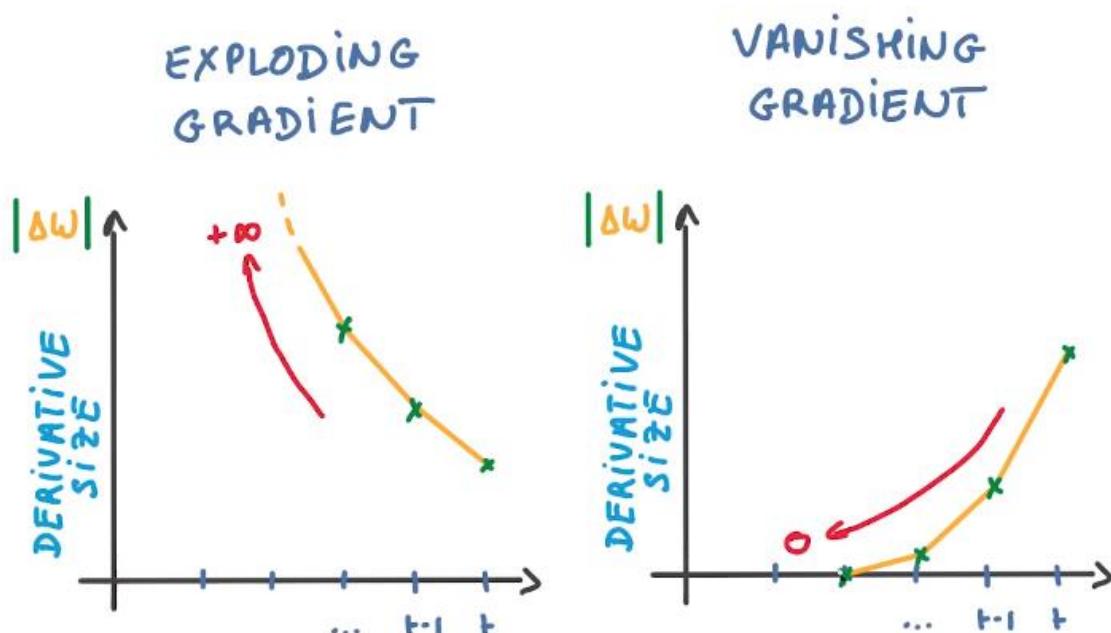
RNN is the artificial neural which were created to analyze and recognize the patterns in the sequences of the data. Due to their internal memory, RNN can certainly remember the things about the inputs they receive.



Most common issues faced with RNN

Although RNN is around for a while and uses backpropagation, there are some common issues faced by developers who work it. Out of all, some of the most common issues are:

- Exploding gradients
- Vanishing gradients



Q3. What is a ResNet, and where would you use it? Is it efficient?

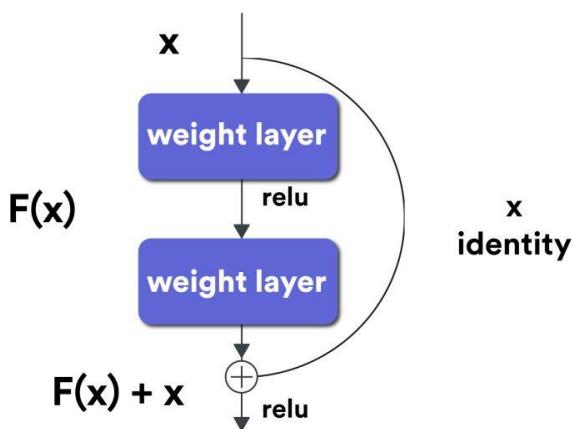
Answer:

Among the various neural networks that are used for computer vision, ResNet (Residual Neural Networks), is one of the most popular ones. It allows us to train extremely deep neural networks, which is the prime reason for its huge usage and popularity. Before the invention of this network, training extremely deep neural networks was almost impossible.

To understand why we must look at the vanishing gradient problem which is an issue that arises when the gradient is backpropagated to all the layers. As a large number of multiplications are performed, the size of the network keeps decreasing till it becomes extremely small, and thus, the network starts performing badly. ResNet helps to counter the vanishing gradient problem.

The efficiency of this network is highly dependent on the concept of skip connections. Skip connections are a method of allowing a shortcut path through which the gradient can flow, which in effect helps counter the vanishing gradient problem.

An example of a skip connection is shown below:



In general, a skip connection allows us to skip the training of a few layers. Skip connections are also called identity shortcut connections as they allow us to directly compute an identity function by just relying on these connections and not having to look at the whole network.

The skipping of these layers makes ResNet an extremely efficient network.

Q4. Transfer learning is one of the most useful concepts today. Where can it be used?

Answer:

Pre-trained models are probably one of the most common use cases for transfer learning.

For anyone who does not have access to huge computational power, training complex models is always a challenge. Transfer learning aims to help by both improving the performance and speeding up your network.

In layman terms, transfer learning is a technique in which a model that has already been trained to do one task is used for another without much change. This type of learning is also called multi-task learning.

Many models that are pre-trained are available online. Any of these models can be used as a starting point in the creation of the new model required. After just using the weights, the model must be refined and adapted on the required data by tuning the parameters of the model.

Model name	Speed (ms)	COCO mAP[^1]	Outputs
ssd_mobilenet_v1_coco	30	21	Boxes
ssd_mobilenet_v1_0.75_depth_coco ☆	26	18	Boxes
ssd_mobilenet_v1_quantized_coco ☆	29	18	Boxes
ssd_mobilenet_v1_0.75_depth_quantized_coco ☆	29	16	Boxes
<u>ssd_mobilenet_v1_ppn_coco ☆</u> 	26	20	Boxes
ssd_mobilenet_v1_fpn_coco ☆	56	32	Boxes
ssd_resnet_50_fpn_coco ☆	76	35	Boxes
ssd_mobilenet_v2_coco	31	22	Boxes
ssd_mobilenet_v2_quantized_coco	29	22	Boxes
ssdlite_mobilenet_v2_coco	27	22	Boxes
ssd_inception_v2_coco	42	24	Boxes
faster_rcnn_inception_v2_coco	58	28	Boxes

The general idea behind transfer learning is to transfer knowledge not data. For humans, this task is easy – we can generalize models that we have mentally created a long time ago for a different purpose. One or two samples is almost always enough. However, in the case of neural networks, a huge amount of data and computational power are required.

Transfer learning should generally be used when we don't have a lot of labeled training data, or if there already exists a network for the task you are trying to achieve, probably trained on a much more massive dataset. Note, however, that the input of the model must have the same size during training. Also, this works only if the tasks are fairly similar to each other, and the features learned can be generalized. For example, something like learning how to recognize vehicles can probably be extended to learn how to recognize airplanes and helicopters.

Q5. What does tuning of hyperparameters signify? Explain with examples.

Answer:

A hyperparameter is just a variable that defines the structure of the network. Let's go through some hyperparameters and see the effect of tuning them.

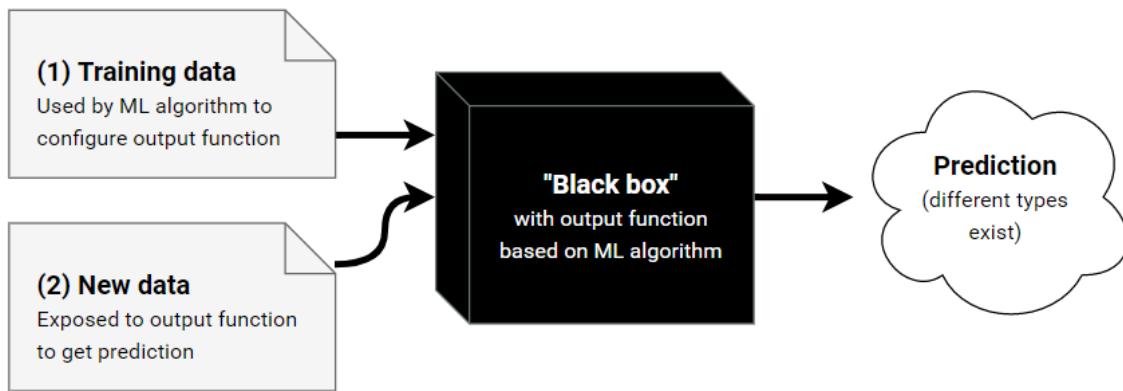
1. A number of hidden layers – Most times, the presence or absence of a large number of hidden layers may determine the output, accuracy and training time of the neural network. Having a large number of these layers may sometimes cause an increase in accuracy.
2. Learning rate – This is simply a measure of how fast the neural network will change its parameters. A large learning rate may lead to the network not being able to converge, but might also speed up learning. On the other hand, a smaller value for the learning rate will probably slow down the network but might lead to the network being able to converge.
3. Number of epochs – This is the number of times the entire training data is run through the network. Increasing the number of epochs leads to better accuracy.
4. Momentum – Momentum is a measure of how and where the network will go while taking into account all of its past actions. A proper measure of momentum can lead to a better network.
5. Batch Size – Batch size determines the number of subsamples that are inputs to the network before every parameter update.

Q6. Why are deep learning models referred as black boxes?

Answer:

Lately, the concept of deep learning being a black box has been floating around. A black box is a system whose functioning cannot be properly grasped, but the output produced can be understood and utilized.

Now, since most models are mathematically sound and are created based on legit equations, how is it possible that we do not know how the system works?



First, it is almost impossible to visualize the functions that are generated by a system. Most machine learning models end up with such complex output that a human can't make sense of it.

Second, there are networks with millions of hyperparameters. As a human, we can grasp around 10 to 15 parameters. But analysing a million of them seems out of the question.

Third and most important, it becomes very hard, if not impossible, to trace back why the system made the decisions it did. This may not sound like a huge problem to worry about but consider the case of a self driving car. If the car hits someone on the road, we need to understand why that happened and prevent it. But this isn't possible if we do not understand how the system works.

To make a deep learning model not be a black box, a new field called Explainable Artificial Intelligence or simply, Explainable AI is emerging. This field aims to be able to create intermediate results and trace back the decision-making process of a system.

Q7. Why do we have gates in neural networks?

Answer:

To understand gates, we must first understand recurrent neural networks.

Recurrent neural networks allow information to be stored as a memory using loops. Thus, the output of a recurrent neural network is not only based on the current input but also the past inputs which are stored in the memory of the network. Backpropagation is done through time, but in general, the truncated version of this is used for longer sequences.

Gates are generally used in networks that are dependent on time. In effect, any network which would require memory, so to speak, would benefit from the use of gates. These gates are generally used to keep track of any information that is required by the network without leading to a state of either vanishing or exploding gradients. Such a network can also preserve the error through time. Since a sense of constant error is maintained, the network can learn better.

Logic Gates

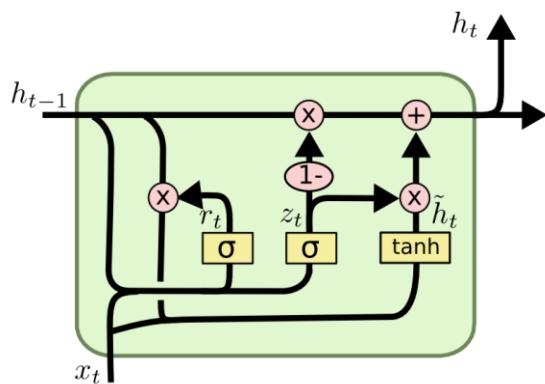
Name	NOT	AND	NAND	OR	NOR	XOR	XNOR																																																																																																
Alg. Expr.	\bar{A}	AB	$\bar{A}\bar{B}$	$A + B$	$\bar{A} + \bar{B}$	$A \oplus B$	$\bar{A} \oplus \bar{B}$																																																																																																
Symbol																																																																																																							
Truth Table	<table border="1"> <tr> <th>A</th><th>X</th></tr> <tr> <td>0</td><td>1</td></tr> <tr> <td>1</td><td>0</td></tr> </table>	A	X	0	1	1	0	<table border="1"> <tr> <th>B</th><th>A</th><th>X</th></tr> <tr> <td>0</td><td>0</td><td>0</td></tr> <tr> <td>0</td><td>1</td><td>0</td></tr> <tr> <td>1</td><td>0</td><td>0</td></tr> <tr> <td>1</td><td>1</td><td>1</td></tr> </table>	B	A	X	0	0	0	0	1	0	1	0	0	1	1	1	<table border="1"> <tr> <th>B</th><th>A</th><th>X</th></tr> <tr> <td>0</td><td>0</td><td>1</td></tr> <tr> <td>0</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>1</td></tr> <tr> <td>1</td><td>1</td><td>1</td></tr> </table>	B	A	X	0	0	1	0	1	1	1	0	1	1	1	1	<table border="1"> <tr> <th>B</th><th>A</th><th>X</th></tr> <tr> <td>0</td><td>0</td><td>1</td></tr> <tr> <td>0</td><td>1</td><td>0</td></tr> <tr> <td>1</td><td>0</td><td>0</td></tr> <tr> <td>1</td><td>1</td><td>0</td></tr> </table>	B	A	X	0	0	1	0	1	0	1	0	0	1	1	0	<table border="1"> <tr> <th>B</th><th>A</th><th>X</th></tr> <tr> <td>0</td><td>0</td><td>0</td></tr> <tr> <td>0</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>1</td></tr> <tr> <td>1</td><td>1</td><td>0</td></tr> </table>	B	A	X	0	0	0	0	1	1	1	0	1	1	1	0	<table border="1"> <tr> <th>B</th><th>A</th><th>X</th></tr> <tr> <td>0</td><td>0</td><td>0</td></tr> <tr> <td>0</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>1</td></tr> <tr> <td>1</td><td>1</td><td>1</td></tr> </table>	B	A	X	0	0	0	0	1	1	1	0	1	1	1	1	<table border="1"> <tr> <th>B</th><th>A</th><th>X</th></tr> <tr> <td>0</td><td>0</td><td>1</td></tr> <tr> <td>0</td><td>1</td><td>0</td></tr> <tr> <td>1</td><td>0</td><td>0</td></tr> <tr> <td>1</td><td>1</td><td>1</td></tr> </table>	B	A	X	0	0	1	0	1	0	1	0	0	1	1	1
A	X																																																																																																						
0	1																																																																																																						
1	0																																																																																																						
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					

These gated units can be considered as units with recurrent connections. They also contain additional neurons, which are gates. If you relate this process to a signal processing system, the gate is used to

regulate which part of the signal passes through. A sigmoid activation function is used which means that the values taken are from 0 to 1.

An advantage of using gates is that it enables the network to either forget information that it has already learned or to selectively ignore information either based on the state of the network or the input the gate receives.

Gates are extensively used in recurrent neural networks, especially in Long Short-Term Memory (LSTM) networks. A general LSTM network will have 3 to 5 gates, typically an input gate, output gate, hidden gate, and activation gate.



$$z_t = \sigma (W_z \cdot [h_{t-1}, x_t])$$

$$r_t = \sigma (W_r \cdot [h_{t-1}, x_t])$$

$$\tilde{h}_t = \tanh (W \cdot [r_t * h_{t-1}, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Q8. What is a Sobel filter?

Answer:

The Sobel filter performs a two-dimensional spatial gradient measurement on a given image, which then emphasizes regions that have a high spatial frequency. In effect, this means finding edges.

In most cases, Sobel filters are used to find the approximate absolute gradient magnitude for every point in a grayscale image. The operator consists of a pair of 3×3 convolution kernels. One of these kernels is rotated by 90 degrees.

-1	0	+1
-2	0	+2
-1	0	+1

x filter

+1	+2	+1
0	0	0
-1	-2	-1

y filter

These kernels respond to edges that run horizontal or vertical with respect to the pixel grid, one kernel for each orientation. A point to note is that these kernels can be applied either separately or can be combined to find the absolute magnitude of the gradient at every point.

The Sobel operator has a large convolution kernel, which ends up smoothing the image to a greater extent, and thus, the operator becomes less sensitive to noise. It also produces higher output values for similar edges compared to other methods.

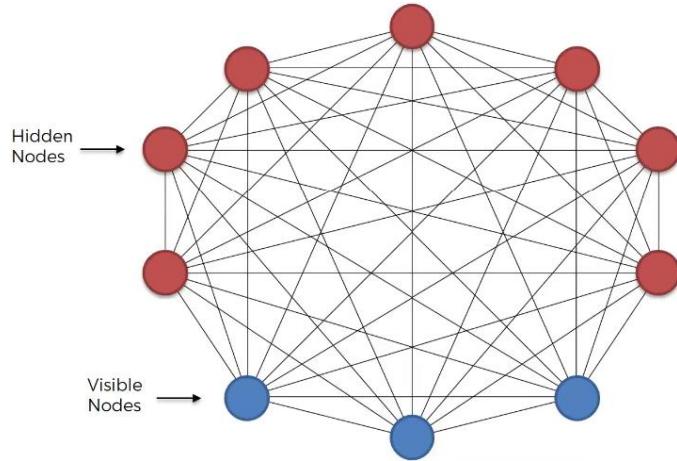
To overcome the problem of output values from the operator overflowing the maximum allowed pixel value per image type, avoid using image types that support pixel values.

Q9. What is the purpose of a Boltzmann Machine?

Answer:

Boltzmann machines are algorithms that are based on physics, specifically thermal equilibrium. A special and more well-known case of Boltzmann machines is the Restricted Boltzmann machine, which is a type of Boltzmann machine where there are no connections between hidden layers of the network.

The concept was coined by Geoff Hinton, who most recently won the Turing award. In general, the algorithm uses the laws of thermodynamics and tries to optimize a global distribution of energy in the system.



In discrete mathematical terms, a restricted Boltzmann machine can be called a symmetric bipartite graph, i.e. two symmetric layers. These machines are a form of unsupervised learning, which means that there are no labels provided with data. It uses stochastic binary units to reach this state.

Boltzmann machines are derived from Markov state machines. A Markov State Machine is a model that can be used to represent almost any computable function. The restricted Boltzmann machine can be regarded as an undirected graphical model. It is used in dimensionality reduction, collaborative filtering, learning features as well as modeling. It can also be used for classification and regression. In general, restricted Boltzmann machines are composed of a two-layer network, which can then be extended further.

Note that these models are probabilistic since each of the nodes present in the system learns low-level features from items in the dataset. For example, if we take a grayscale image, each node that is responsible for the visible layer will take just one-pixel value from the image.

A part of the process of creating such a machine is a feature hierarchy where sequences of activations are grouped in terms of features. In thermodynamics principles, simulated annealing is a process that the machine follows to separate signal and noise.

Q10. What are the types of weight initialization?

Answer:

There are two major types of weight initialization:- zero initialization and random initialization.

Zero initialization: In this process, biases and weights are initialised to 0. If the weights are set to 0, all derivatives with respect to the loss functions in the weight matrix become equal. Hence, none of the weights change during subsequent iterations. Setting the bias to 0 cancels out any effect it may have.

All hidden units become symmetric due to zero initialization. In general, zero initialization is not very useful or accurate for classification and thus must be avoided when any classification task is required.

Random initialization: As compared to 0 initialization, this involves setting random values for the weights. The only disadvantage is that set very high values will increase the learning time as the sigmoid activation function maps close to 1. Likewise, if low values are set, the learning time increases as the activation function is mapped close to 0.

Setting too high or too low values thus generally leads to the exploding or vanishing gradient problem.

New types of weight initialization like “**He initialization**” and “**Xavier initialization**” have also emerged. These are based on specific equations and are not mentioned here due to their sheer complexity.

Q1. Where is the confusion matrix used? Which module would you use to show it?

Answer:

In machine learning, confusion matrix is one of the easiest ways to summarize the performance of your algorithm.

At times, it is difficult to judge the accuracy of a model by just looking at the accuracy because of problems like unequal distribution. So, a better way to check how good your model is, is to use a confusion matrix.

First, let's look at some key terms.

Classification accuracy – This is the ratio of the number of correct predictions to the number of predictions made

True positives – Correct predictions of true events

False positives – Incorrect predictions of true events

True negatives – Correct predictions of false events

False negatives – Incorrect predictions of false events.

The confusion matrix is now simply a matrix containing true positives, false positives, true negatives, false negatives.

		Predicted Class		
		Positive	Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	Sensitivity $\frac{TP}{(TP + FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	Specificity $\frac{TN}{(TN + FP)}$
	Precision $\frac{TP}{(TP + FP)}$	Negative Predictive Value $\frac{TN}{(TN + FN)}$	Accuracy $\frac{TP + TN}{(TP + TN + FP + FN)}$	

Q2: What is Accuracy?

Answer:

It is the most intuitive performance measure and it simply a ratio of correctly predicted to the total observations. We can say as, if we have high accuracy, then our model is best. Yes, we could say that accuracy is a great measure but only when you have symmetric datasets where false positives and false negatives are almost same.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{False Negative} + \text{True Negative}}$$

		Condition Absent	Condition Present
Negative Result	True Negative	False Negative	
	False Positive	True Positive	

Q3: What is Precision?

Answer:

It is also called as the positive predictive value. Number of correct positives in your model that predicts compared to the total number of positives it predicts.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Precision} = \frac{\text{True Positives}}{\text{Total predicted positive}}$$

It is the number of positive elements predicted properly divided by the total number of positive elements predicted.

We can say Precision is a measure of exactness, quality, or accuracy. High precision

Means that more or all of the positive results you predicted are correct.

Q4: What is Recall?

Answer:

Recall we can also called as sensitivity or true positive rate.

It is several positives that our model predicts compared to the actual number of positives in our data.

$$\text{Recall} = \text{True Positives} / (\text{True Positives} + \text{False Positives})$$

$$\text{Recall} = \text{True Positives} / \text{Total Actual Positive}$$

Recall is a measure of completeness. High recall which means that our model classified most or all of the possible positive elements as positive.

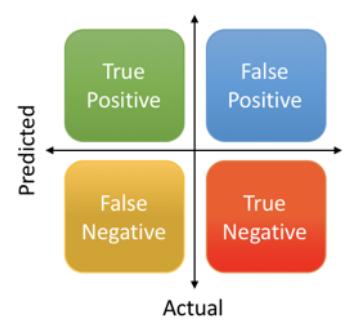
Q5: What is F1 Score?

Answer:

We use Precision and recall together because they complement each other in how they describe the effectiveness of a model. The F1 score that combines these two as the weighted harmonic mean of precision and recall.

$$\text{F1 Score} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

$$\begin{aligned}\text{Precision} &= \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \\ \text{Recall} &= \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \\ \text{Accuracy} &= \frac{\text{True Positive} + \text{True Negative}}{\text{Total}}\end{aligned}$$



Q6: What is Bias and Variance trade-off?

Answer:

Bias

Bias means it's how far are the predicted values from the actual values. If the average predicted values are far off from the actual values, then we call this one as having high bias.

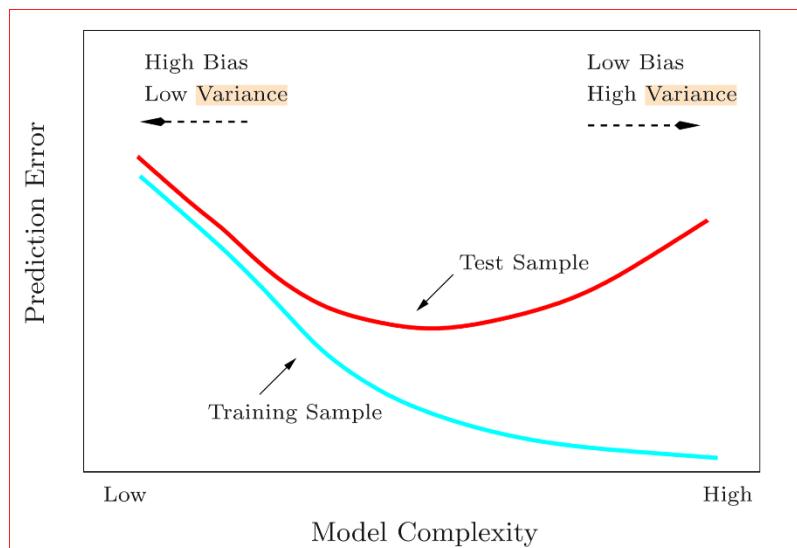
When our model has a high bias, then it means that our model is too simple and does not capture the complexity of data, thus underfitting the data.

Variance

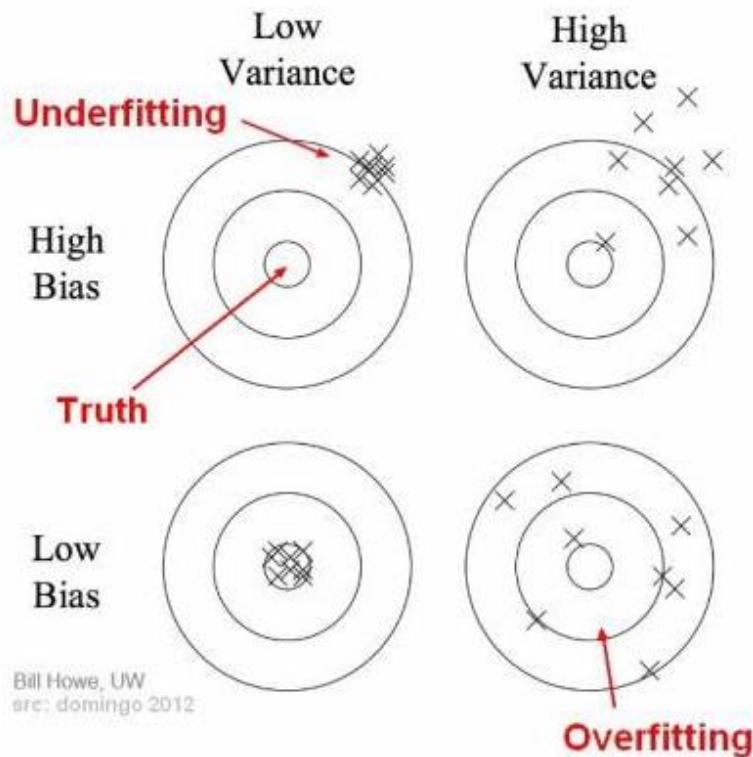
It occurs when our model performs well on the trained dataset but does not do well on a dataset that it is not trained on, like a test dataset or validation dataset. It tells us that actual value is how much scattered from the predicted value.

Because of High variance it causes overfitting that implies that the algorithm models random noise present in the training data.

When model has high variance, then model becomes very flexible and tunes itself to the data points of the training set.



Bias-variance: It decomposition essentially decomposes the learning error from any algorithm by adding bias, the variance and a bit of irreducible error due to noise in the underlying dataset. Essentially, if we make the model more complex and add more variables, We'll lose bias but gain some variance —to get the optimally reduced amount of error, you'll have to tradeoff bias and variance. We don't want either high bias or high variance in your model.



Bias and variance using bulls-eye diagram

Q7. What is data wrangling? Mention three points to consider in the process.

Answer:

Data wrangling is a process by which we convert and map data. This changes data from its raw form to a format that is a lot more valuable.

Data wrangling is the first step for machine learning and deep learning. The end goal is to provide data that is actionable and to provide it as fast as possible.

There are three major things to focus on while talking about data wrangling –

1. Acquiring data

The first and probably the most important step in data science is the acquiring, sorting and cleaning of data. This is an extremely tedious process and requires the most amount of time.

One needs to:

- Check if the data is valid and up-to-date.
- Check if the data acquired is relevant for the problem at hand.

Sources for data collection Data is publicly available on various websites like kaggle.com, [data.gov](#), [World Bank](#), [Five Thirty Eight Datasets](#), AWS Datasets, Google Datasets.

2. Data cleaning

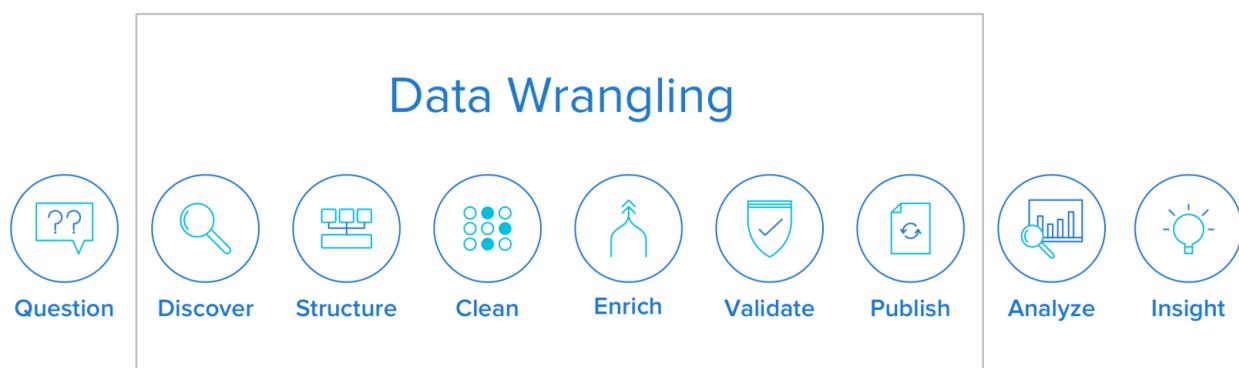
Data cleaning is an essential component of data wrangling and requires a lot of patience. To make the job easier it is first essential to format the data make the data readable for humans at first.

The essentials involved are:

- Format the data to make it more readable
- Find outliers (data points that do not match the rest of the dataset) in data
- Find missing values and remove them from the data set (without this, any model being trained becomes incomplete and useless)

3. Data Computation

At times, your machine not have enough resources to run your algorithm e.g. you might not have a GPU. In these cases, you can use publicly available APIs to run your algorithm. These are standard end points found on the web which allow you to use computing power over the web and process data without having to rely on your own system. An example would be the Google Colab Platform.



Q8. Why is normalization required before applying any machine learning model? What module can you use to perform normalization?

Answer:

Normalization is a process that is required when an algorithm uses something like distance measures. Examples would be clustering data, finding cosine similarities, creating recommender systems.

Normalization is not always required and is done to prevent variables that are on higher scale from affecting outcomes that are on lower levels. For example, consider a dataset of employees' income. This data won't be on the same scale if you try to cluster it. Hence, we would have to normalize the data to prevent incorrect clustering.

A key point to note is that normalization does not distort the differences in the range of values.

A problem we might face if we don't normalize data is that gradients would take a very long time to descend and reach the global maxima/ minima.

For numerical data, normalization is generally done between the range of 0 to 1.

The general formula is:

$$X_{\text{new}} = (x - \text{xmin}) / (\text{xmax} - \text{xmin})$$

Normalization Formula

$$X_{\text{normalized}} = \frac{(X - X_{\text{minimum}})}{(X_{\text{maximum}} - X_{\text{minimum}})}$$



Q9. What is the difference between feature selection and feature extraction?

Feature selection and feature extraction are two major ways of fixing the curse of dimensionality

1. Feature selection:

Feature selection is used to filter a subset of input variables on which the attention should focus. Every other variable is ignored. This is something which we, as humans, tend to do subconsciously.

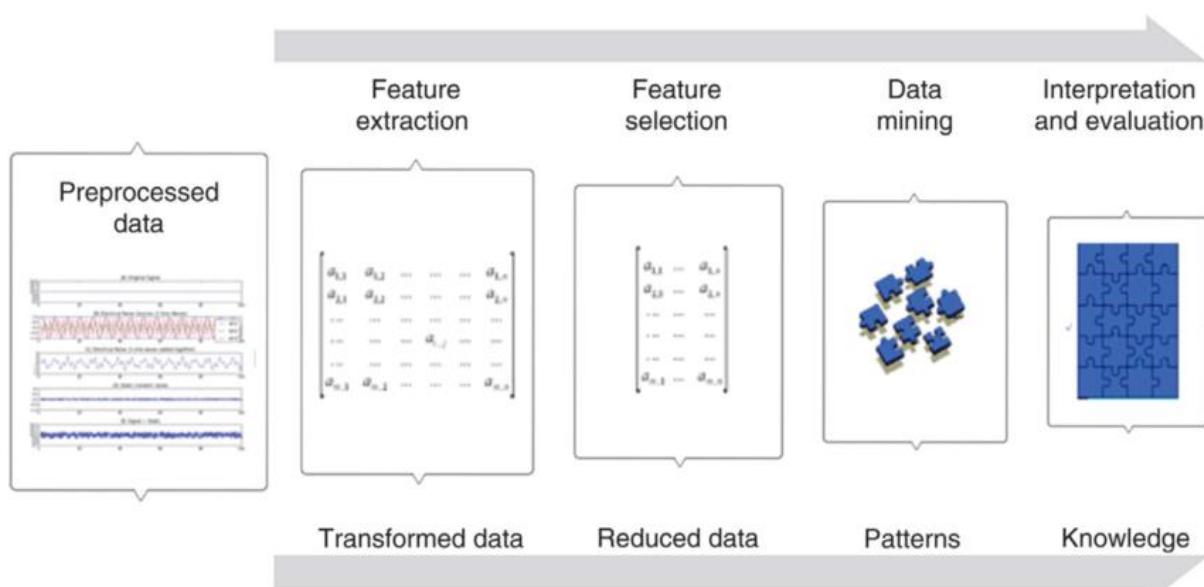
Many domains have tens of thousands of variables out of which most are irrelevant and redundant. Feature selection limits the training data and reduces the amount of computational resources used. It can significantly improve a learning algorithms performance.

In summary, we can say that the goal of feature selection is to find out an optimal feature subset. This might not be entirely accurate, however, methods of understanding the importance of features also exist. Some modules in python such as Xgboost help achieve the same.

2. Feature extraction

Feature extraction involves transformation of features so that we can extract features to improve the process of feature selection. For example, in an unsupervised learning problem, the extraction of bigrams from a text, or the extraction of contours from an image are examples of feature extraction.

The general workflow involves applying feature extraction on given data to extract features and then apply feature selection with respect to the target variable to select a subset of data. In effect, this helps improve the accuracy of a model.



Q10. Why is polarity and subjectivity an issue?

Polarity and subjectivity are terms which are generally used in sentiment analysis.

Polarity is the variation of emotions in a sentence. Since sentiment analysis is widely dependent on emotions and their intensity, polarity turns out to be an extremely important factor.

In most cases, opinions and sentiment analysis are evaluations. They fall under the categories of emotional and rational evaluations.

Rational evaluations, as the name suggests, are based on facts and rationality while emotional evaluations are based on non-tangible responses, which are not always easy to detect.

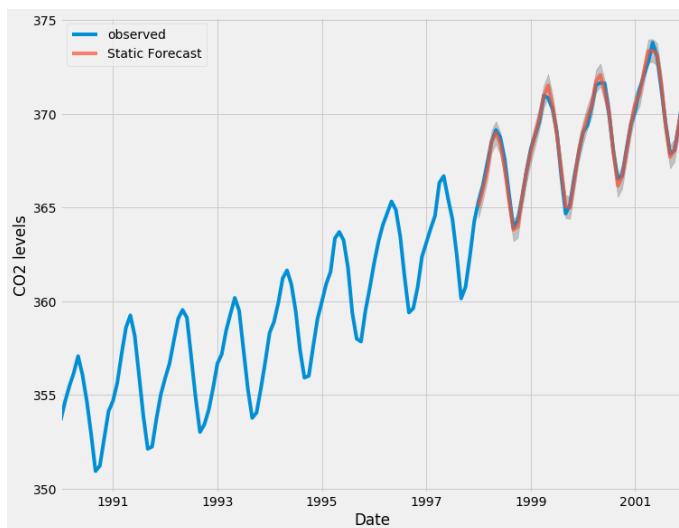
Subjectivity in sentiment analysis, is a matter of personal feelings and beliefs which may or may not be based on any fact. When there is a lot of subjectivity in a text, it must be explained and analysed in context. On the contrary, if there was a lot of polarity in the text, it could be expressed as a positive, negative or neutral emotion.

Q11. When would you use ARIMA?

Answer:

ARIMA is a widely used statistical method which stands for Auto Regressive Integrated Moving Average. It is generally used for analyzing time series data and time series forecasting. Let's take a quick look at the terms involved.

Auto Regression is a model that uses the relationship between the observation and some numbers of lagging observations.



Integrated means use of differences in raw observations which help make the time series stationary.

Moving Averages is a model that uses the relationship and dependency between the observation and residual error from the models being applied to the lagging observations.

Note that each of these components are used as parameters. After the construction of the model, a linear regression model is constructed.

Data is prepared by:

- Finding out the differences
- Removing trends and structures that will negatively affect the model
- Finally, making the model stationary.

Q1. What is Autoregression?

Answer:

The autoregressive (AR) model is commonly used to model time-varying processes and solve problems in the fields of natural science, economics and finance, and others. The models have always been discussed in the context of random process and are often perceived as statistical tools for time series data.

A regression model, like linear regression, models an output value which are based on a linear combination of input values.

Example: $\hat{y} = b_0 + b_1 \cdot X_1$

Where \hat{y} is the prediction, b_0 and b_1 are coefficients found by optimising the model on training data, and X is an input value.

This model technique can be used on the time series where input variables are taken as observations at previous time steps, called lag variables.

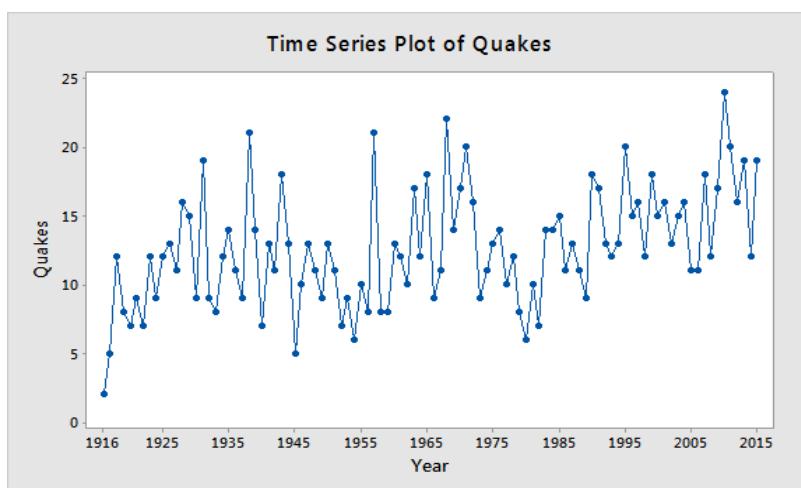
For example, we can predict the value for the next time step ($t+1$) given the observations at the last two time steps ($t-1$ and $t-2$). As a regression model, this would look as follows:

$$X(t+1) = b_0 + b_1 \cdot X(t-1) + b_2 \cdot X(t-2)$$

Because the regression model uses the data from the same input variable at previous time steps, it is referred to as an autoregression.

The notation $AR(p)$ refers to the autoregressive model of order p . The $AR(p)$ model is written

$$X_t = c + \sum_{i=1}^p \varphi_i X_{t-i} + \varepsilon_t.$$



Q2. What is Moving Average?

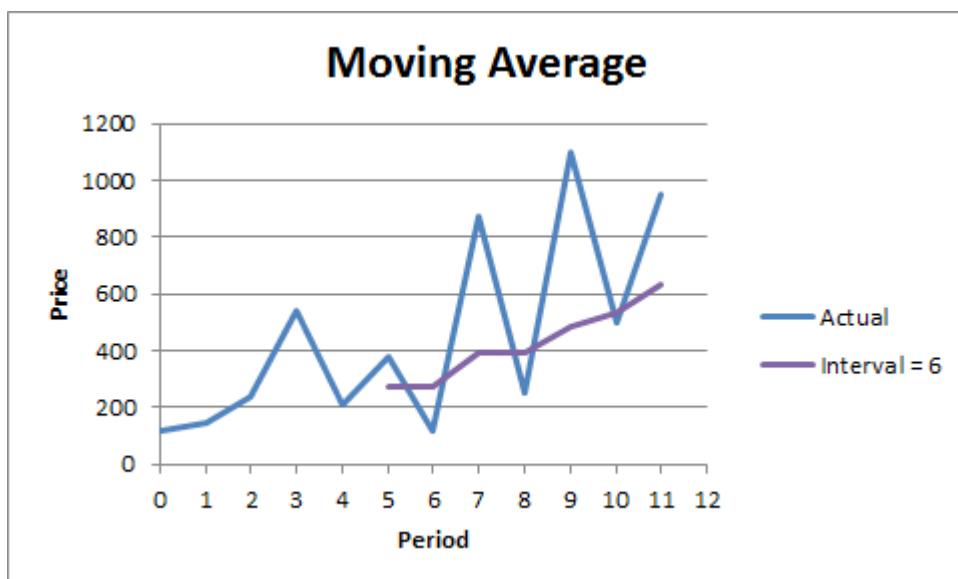
Answer:

Moving average: From a dataset, we will get an overall idea of trends by this technique; it is an average of any subset of numbers. For forecasting long-term trends, the moving average is extremely useful for it. We can calculate it for any period. For example: if we have sales data for twenty years, we can calculate the five-year moving average, a four-year moving average, a three-year moving average and so on. Stock market analysts will often use a 50 or 200-day moving average to help them see trends in the stock market and (hopefully) forecast where the stocks are headed.

The notation $MA(q)$ refers to the moving average model of order q :

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^q \theta_i \varepsilon_{t-i}$$

where the $\theta_1, \dots, \theta_q$ are the parameters of the model, μ is the expectation of X_t (often assumed to equal 0), and the $\varepsilon_t, \varepsilon_{t-1}, \dots$ are again, white noise error terms.



The notation $MA(q)$ refers to the moving average model of order q :

Q3. What is Autoregressive Moving Average (ARMA)?

Answer:

ARMA: It is a model of forecasting in which the methods of autoregression (AR) analysis and moving average (MA) are both applied to time-series data that is well behaved. In ARMA it is assumed that the time series is stationary and when it fluctuates, it does so uniformly around a particular time.

AR (Autoregression model)-

Autoregression (AR) model is commonly used in current spectrum estimation.

The following is the procedure for using ARMA.

- Selecting the AR model and then equalizing the output to equal the signal being studied if the input is an impulse function or the white noise. It should at least be good approximation of signal.
- Finding a model's parameters number using the known autocorrelation function or the data .
- Using the derived model parameters to estimate the power spectrum of the signal.

Moving Average (MA) model-

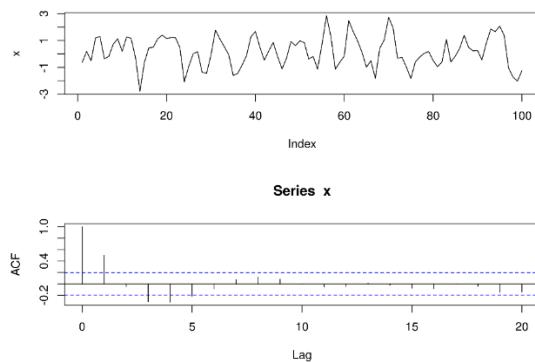
It is a commonly used model in the modern spectrum estimation and is also one of the methods of the model parametric spectrum analysis. The procedure for estimating MA model's signal spectrum is as follows.

- Selecting the MA model and then equalising the output to equal the signal under study in the case where the input is an impulse function or white noise. It should be at least a good approximation of the signal.
- Finding the model's parameters using the known autocorrelation function.
- Estimating the signal's power spectrum using the derived model parameters.

In the estimation of the ARMA parameter spectrum, the AR parameters are first estimated, and then the MA parameters are estimated based on these AR parameters. The spectral estimates of the ARMA model are then obtained. The parameter estimation of the MA model is, therefore often calculated as a process of ARMA parameter spectrum association.

The notation ARMA(p, q) refers to the model with p autoregressive terms and q moving-average terms. This model contains the AR(p) and MA(q) models,

$$X_t = c + \varepsilon_t + \sum_{i=1}^p \varphi_i X_{t-i} + \sum_{i=1}^q \theta_i \varepsilon_{t-i}.$$



Q4. What is Autoregressive Integrated Moving Average (ARIMA)?

Answer:

ARIMA: It is a statistical analysis model that uses time-series data to either better understand the data set or to predict future trends.

An ARIMA model can be understood by the outlining each of its components as follows-

- **Autoregression (AR):** It refers to a model that shows a changing variable that regresses on its own lagged, or prior, values.
- **Integrated (I):** It represents the differencing of raw observations to allow for the time series to become stationary, i.e., data values are replaced by the difference between the data values and the previous values.
- **Moving average (MA):** It incorporates the dependency between an observation and the residual error from the moving average model applied to the lagged observations.

Each component functions as the parameter with a standard notation. For ARIMA models, the standard notation would be the ARIMA with p, d, and q, where integer values substitute for the parameters to indicate the type of the ARIMA model used. The parameters can be defined as-

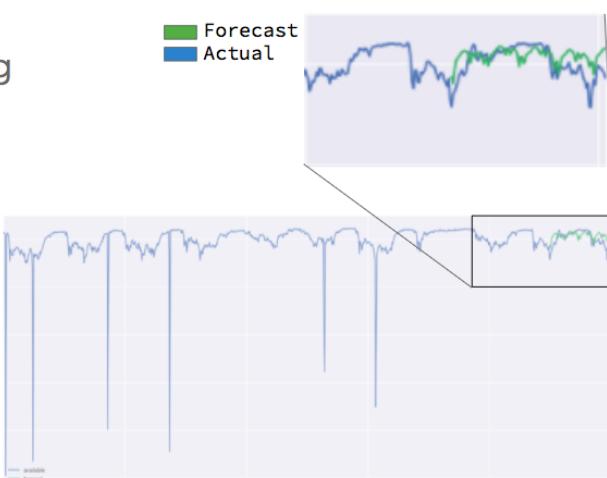
- p : It the number of lag observations in the model; also known as the lag order.
- d : It the number of times that the raw observations are differenced; also known as the degree of differencing.
- q : It the size of the moving average window; also known as the order of the moving average.

ARIMA

Autoregressive Moving Averages

$ARIMA(p,d,q)(P,D,Q)m$

- p = non-seasonal AR order
- d = non-seasonal differencing
- q = non-seasonal MA order
- P = seasonal AR order
- D = seasonal differencing
- Q = seasonal MA order
- m = number of periods/season



$ARIMA(1,0,1)(4, 0, 7, 24)$

Q5.What is SARIMA (Seasonal Autoregressive Integrated Moving-Average)?

Answer:

Seasonal ARIMA: It is an extension of ARIMA that explicitly supports the univariate time series data with the seasonal component.

It adds three new hyper-parameters to specify the autoregression (AR), differencing (I) and the moving average (MA) for the seasonal component of the series, as well as an additional parameter for the period of the seasonality.

Configuring the SARIMA requires selecting hyperparameters for both the trend and seasonal elements of the series.

Trend Elements

Three trend elements require the configuration.

They are same as the ARIMA model, specifically-

p: It is Trend autoregression order.

d: It is Trend difference order.

q: It is Trend moving average order.

Seasonal Elements-

Four seasonal elements are not the part of the ARIMA that must be configured, they are-

P: It is Seasonal autoregressive order.

D: It is Seasonal difference order.

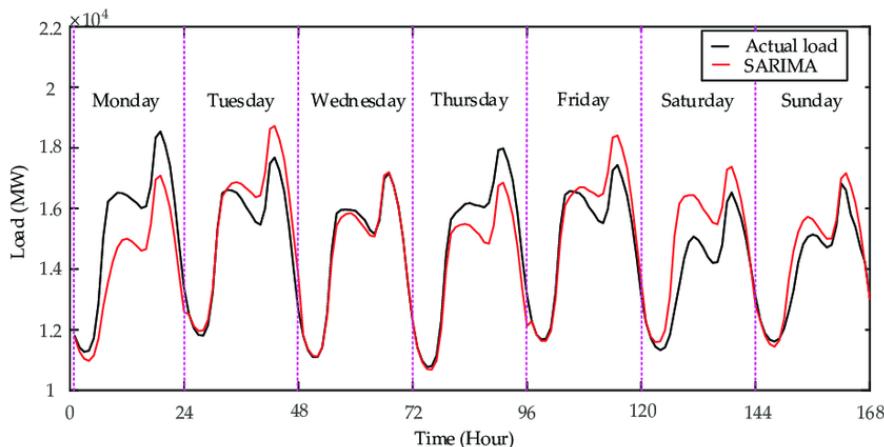
Q: It is Seasonal moving average order.

m: It is the number of time steps for the single seasonal period.

Together, the notation for the SARIMA model is specified as-

SARIMA(p,d,q)(P,D,Q)m-

The elements can be chosen through careful analysis of the ACF and PACF plots looking at the correlations of recent time steps.



Q6. What is Seasonal Autoregressive Integrated Moving-Average with Exogenous Regressors (SARIMAX) ?

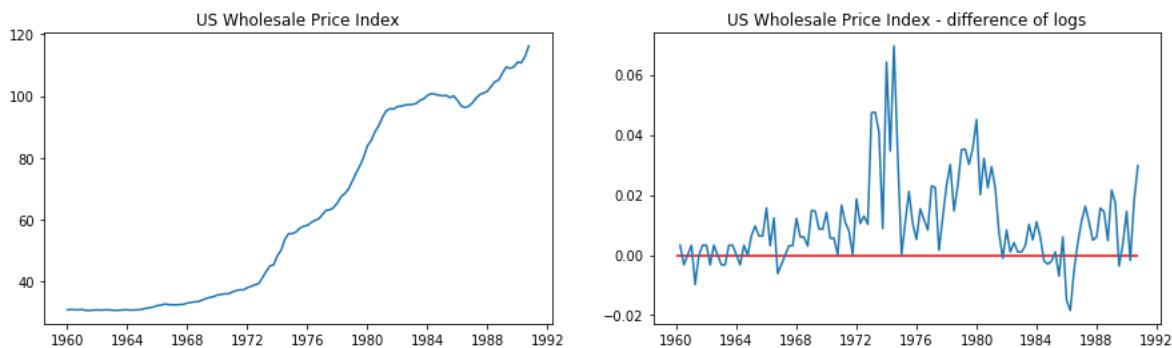
Answer:

SARIMAX: It is an extension of the SARIMA model that also includes the modelling of the exogenous variables.

Exogenous variables are also called the covariates and can be thought of as parallel input sequences that have observations at the same time steps as the original series. The primary series may be referred as endogenous data to contrast it from exogenous sequence(s). The observations for exogenous variables are included in the model directly at each time step and are not modeled in the same way as the primary endogenous sequence (e.g. as an AR, MA, etc. process).

The SARIMAX method can also be used to model the subsumed models with exogenous variables, such as ARX, MAX, ARMAX, and ARIMAX.

The method is suitable for univariate time series with trend and/or seasonal components and exogenous variables.



Q7. What is Vector autoregression (VAR)?

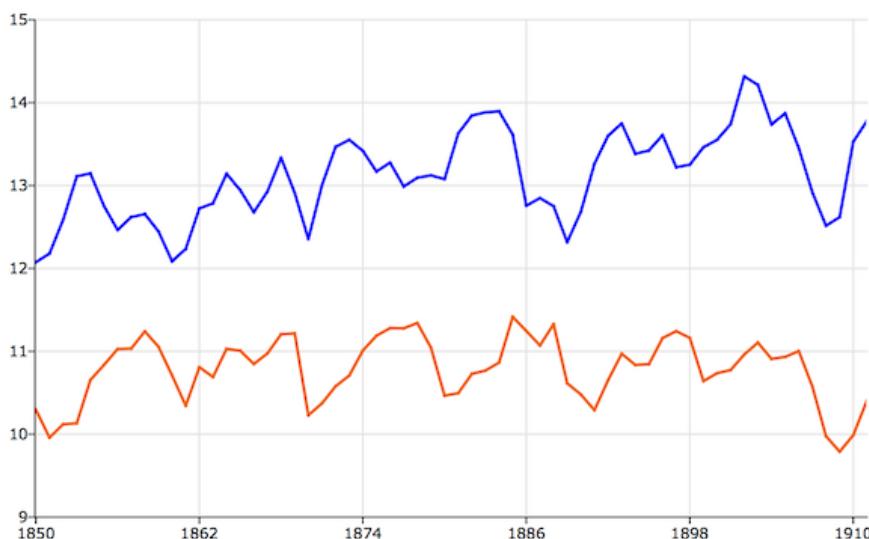
Answer:

VAR: It is a stochastic process model used to capture the linear interdependencies among multiple time series. VAR models generalise the univariate autoregressive model (AR model) by allowing for more than one evolving variable. All variables in the VAR enter the model in the same way: each variable has an equation explaining its evolution based on its own lagged values, the lagged values of the other model variables, and an error term. VAR modelling does not require as much knowledge about the forces influencing the variable as do structural models with simultaneous equations: The only prior knowledge required is a list of variables which can be hypothesised to affect each other intertemporally.

A VAR model describes the evolution of the set of k variables over the same sample period ($t = 1, \dots, T$) as the linear function of only their past values. The variables are collected in the k -vector ($(k \times 1)$ -matrix) y_t , which has as the (i^{th}) element, $y_{i,t}$, the observation at time t of the (i^{th}) variable. Example: if the (i^{th}) variable is the GDP, then $y_{i,t}$ is the value of GDP at time “ t ”.

$$y_t = c + A_1 y_{t-1} + A_2 y_{t-2} + \dots + A_p y_{t-p} + e_t,$$

where the observation y_{t-i} is called the (i -th) **lag** of y , c is the k -vector of constants (intercepts), A_i is a time-invariant ($k \times k$)-matrix, and e_t is a k -vector of error terms satisfying.



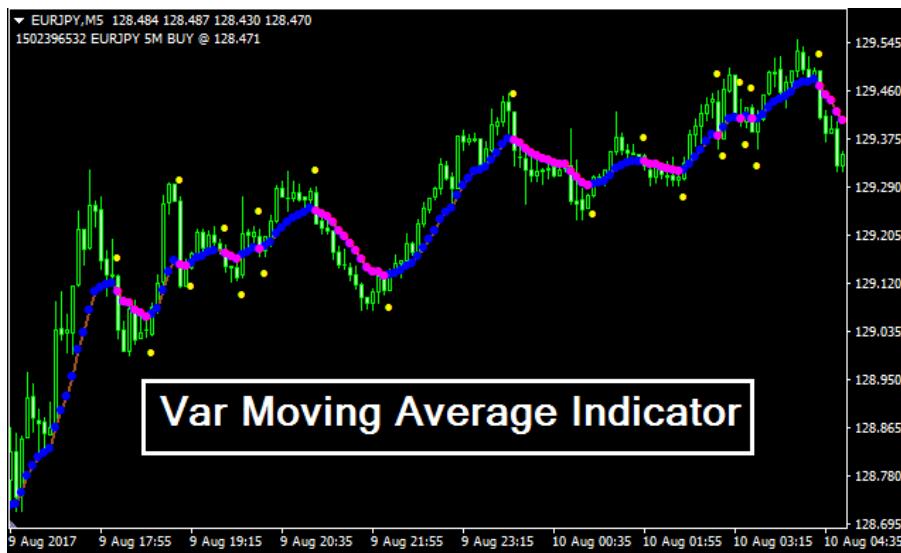
Q8. What is Vector Autoregression Moving-Average (VARMA)?

Answer:

VARMA: It is a method that models the next step in each time series using an ARMA model. It is the generalisation of ARMA to multiple parallel time series. Example- multivariate time series.

The notation for a model involves specifying the order for the AR(p) and the MA(q) models as parameters to the VARMA function, e.g. VARMA (p, q). The VARMA model can also be used to develop VAR or VMA models.

This method is suitable for multivariate time series without trend and seasonal components.



Q9. What is Vector Autoregression Moving-Average with Exogenous Regressors (VARMAX)?

Answer:

VARMAX: It is an extension of the VARMA model that also includes the modelling of the exogenous variables. It is the multivariate version of the ARMAX method.

Exogenous variables are also called the covariates and can be thought of as parallel input sequences that have observations at the same time steps as the original series. The primary series(es) are referred as the endogenous data to contrast it from the exogenous sequence(s). The observations for the exogenous variables are included in the model directly at each time step and are not modeled in the same way as the primary endogenous sequence (Example- as an AR, MA, etc.).

This method can also be used to model subsumed models with exogenous variables, such as VARX and the VMAX.

This method is suitable for multivariate time series without trend and seasonal components and exogenous variables.

Q10. What is Simple Exponential Smoothing (SES)?

Answer:

SES: It method models the next time step as an exponentially weighted linear function of observations at prior time steps.

This method is suitable for univariate time series without trend and seasonal components.

Exponential smoothing is the rule of thumb technique for smoothing time series data using the exponential window function. Whereas in the simple moving average, the past observations are weighted equally, exponential functions are used to assign exponentially decreasing weights over time. It is easily learned and easily applied procedure for making some determination based on prior assumptions by the user, such as seasonality. Exponential smoothing is often used for analysis of time-series data.

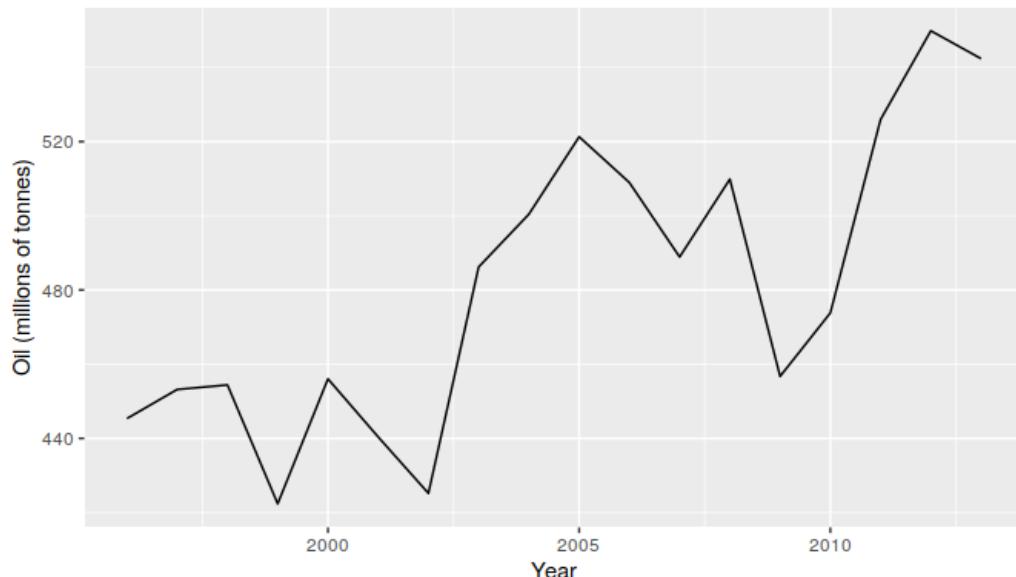
Exponential smoothing is one of many window functions commonly applied to smooth data in signal processing, acting as low-pass filters to remove high-frequency noise.

The raw data sequence is often represented by $\{x_t\}$ beginning at time $t = 0$, and the output of the exponential smoothing algorithm is commonly written as $\{s_t\}$ which may be regarded as a best estimate of what the next value of x will be. When the sequence of observations begins at time $t=0$, the simplest form of exponential smoothing is given by the formulas:

$$s_0 = x_0$$

$$s_t = \alpha x_t + (1 - \alpha)s_{t-1}, \quad t > 0$$

where α is the *smoothing factor*, and $0 < \alpha < 1$.



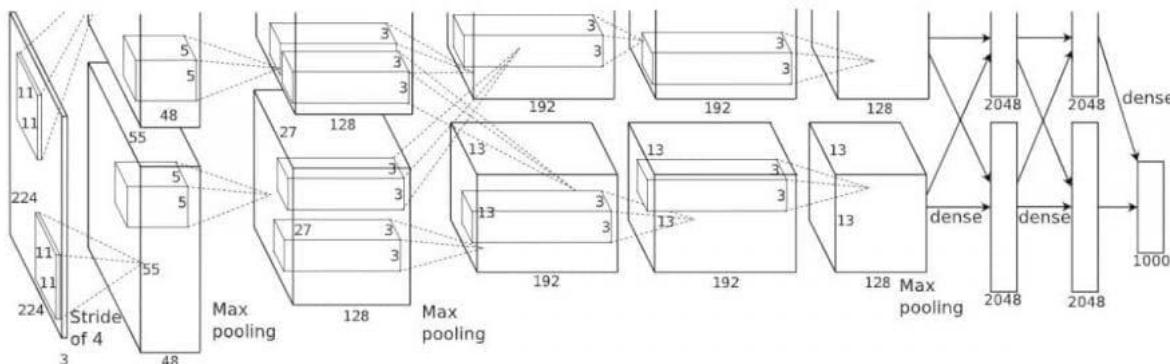
Q1. What is Alexnet?

Answer:

The Alex Krizhevsky, Geoffrey Hinton and Ilya Sutskever created the neural network architecture called ‘AlexNet’ and won Image Classification Challenge (ILSVRC) in 2012. They trained their network on 1.2 million high-resolution images into 1000 different classes with 60 million parameters and 650,000 neurons. The training was done on two GPUs with split layer concept because GPUs were a little bit slow at that time.

AlexNet is the name of convolutional neural network which has had a large impact on the field of machine learning, specifically in the application of deep learning to machine vision. The network had very similar architecture as the LeNet by Yann LeCun et al. but was deeper with more filters per layer, and with the stacked convolutional layers. It consists of (11×11 , 5×5 , 3×3 , convolutions), max pooling, dropout, data augmentation, ReLU activations and SGD with the momentum. It attached with ReLU activations after every convolutional and fully connected layer. AlexNet was trained for six days simultaneously on two Nvidia Geforce GTX 580 GPUs, which is the reason for why their network is split into the two pipelines.

Architecture



AlexNet contains eight layers with weights, first five are convolutional, and the remaining three are fully connected. The output of last fully-connected layer is fed to a 1000-way softmax which produces a distribution over the 1000 class labels. The network maximises the multinomial logistic regression objective, which is equivalent to maximising the average across training cases of the log-probability of the correct label under the prediction distribution. The kernels of second, fourth, and the fifth convolutional layers are connected only with those kernel maps in the previous layer which reside on the same GPU. The kernels of third convolutional layer are connected to all the kernel maps in second layer. The neurons in fully connected layers are connected to all the neurons in the previous layers.

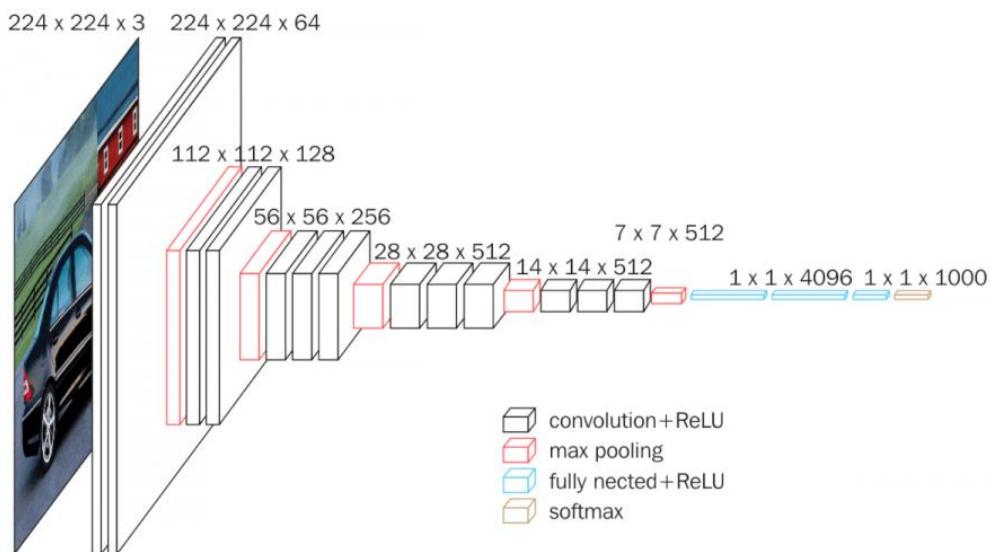
In short, AlexNet contains five convolutional layers and three fully connected layers. Relu is applied after the very convolutional and the fully connected layer. Dropout is applied before the first and second fully connected year. The network has the 62.3 million parameters and needs 1.1 billion computation units in a forward pass. We can also see convolution layers, which accounts for 6% of all the parameters, consumes 95% of the computation.

Q2. What is VGGNet?

Answer:

VGGNet consists of 16 convolutional layers and is very appealing because of its very uniform architecture. Similar to AlexNet, only 3x3 convolutions, but lots of filters. Trained on 4 GPUs for 2–3 weeks. It is currently the most preferred choice in the community for extracting features from images. The weight configuration of the VGGNet is publicly available and has been used in many other applications and challenges as a baseline feature extractor. However, VGGNet consists of 138 million parameters, which can be a bit challenging to handle.

There are multiple variants of the VGGNet (VGG16, VGG19 etc.) which differ only in total number of layers in the networks. The structural details of the VGG16 network has been shown:



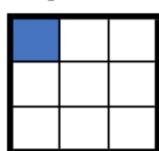
The idea behind having the fixed size kernels is that all the variable size convolutional kernels used in the Alexnet (11x11, 5x5, 3x3) can be replicated by making use of multiple 3x3 kernels as the building blocks. The replication is in term of the receptive field covered by kernels .

Let's consider the example. Say we have an input layer of the size $5 \times 5 \times 1$. Implementing the conv layer with kernel size of 5×5 and stride one will the results and output feature map of (1×1) . The same output feature map can be obtained by implementing the two (3×3) Conv layers with stride of 1 as below:

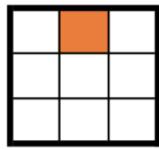
Input Feature Map
and Receptive Field

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

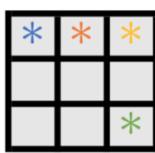
Output for each receptive field



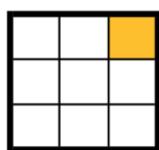
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25



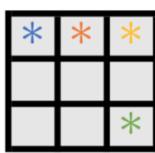
Output Feature Map of 1st conv layer



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25



Input Feature Map of 2nd conv layer



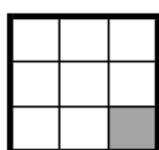
Output Feature Map of 2nd conv layer

•

•

•

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

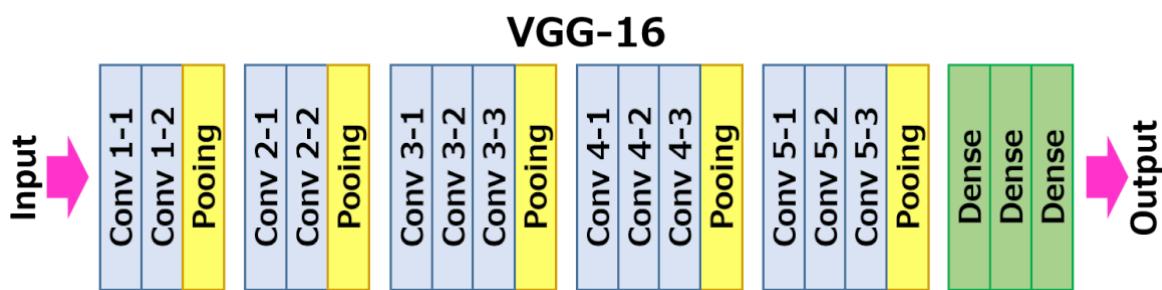


Now, let's look at the number of the variables needed to be trained. For a 5×5 Conv layer filter, the number of variables is 25. On the other hand, two conv layers of kernel size 3×3 have a total of $3 \times 3 \times 2 = 18$ variables (a reduction of 28%).

Q3. What is VGG16?

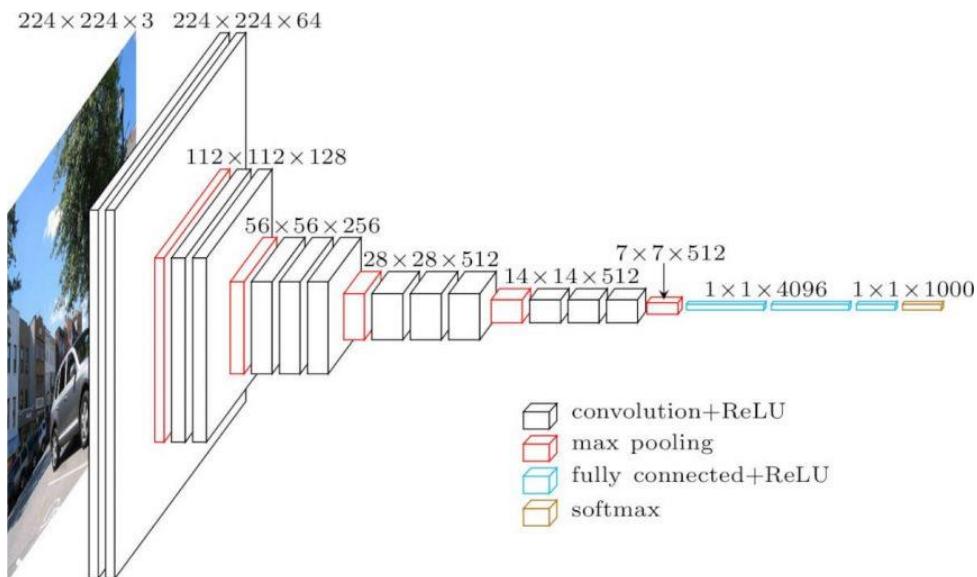
Answer:

VGG16: It is a convolutional neural network model proposed by the K. Simonyan and A. Zisserman from the University of Oxford in the paper “Very Deep Convolutional Networks for the Large-Scale Image Recognition”. The model achieves 92.7% top 5 test accuracy in ImageNet, which is the dataset of over 14 million images belonging to the 1000 classes. It was one of famous model submitted to ILSVRC-2014. It improves AlexNet by replacing the large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. VGG16 was trained for weeks and was using NVIDIA Titan Black GPU’s.



The Architecture

The architecture depicted below is VGG16.



The input to the Cov1 layer is of fixed size of 224 x 224 RGB image. The image is passed through the stack of convolutional (conv.) layers, where the filters were used with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, centre). In one of the configurations, it also utilises the 1×1 convolution filters, which can be seen as the linear transformation of the input channels . The convolution stride is fixed to the 1 pixel, the spatial padding of the Conv. layer input is such that, the spatial resolution is preserved after the convolution, i.e. the

padding is 1-pixel for 3×3 Conv. layers. Spatial pooling is carried out by the five max-pooling layers, which follows some of the Conv. Layers. Max-pooling is performed over the 2×2 pixel window, with stride 2.

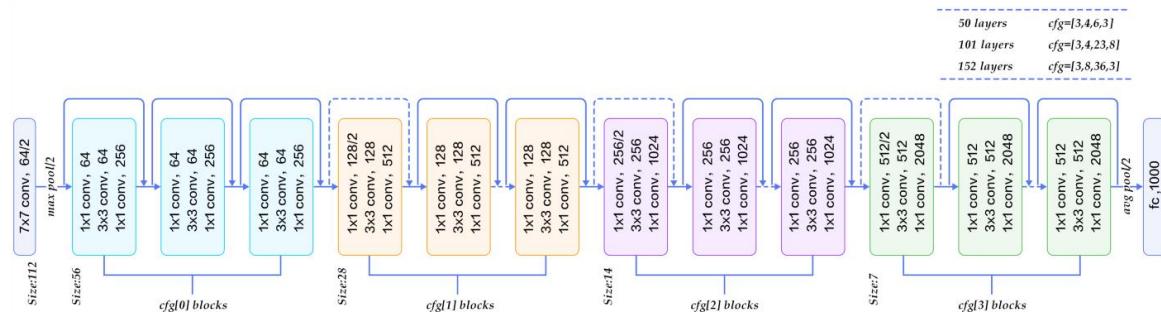
Three Fully-Connected (FC) layers follow the stack of convolutional layers (which has the different depth in different architectures): the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels . The final layer is softmax layer. The configurations of the fully connected layers is same in all the networks.

All hidden layers are equipped with rectification (ReLU) non-linearity. It is also noted that none of the networks (except for one) contain the Local Response Normalisation (LRN), such normalisation does not improve the performance on the ILSVRC dataset, but leads to increased memory consumption and computation time.

Q4. What is ResNet?

Answer:

At the ILSVRC 2015, so-called Residual Neural Network (ResNet) by the Kaiming He et al introduced the anovel architecture with “skip connections” and features heavy batch normalisation. Such skip connections are also known as the gated units or gated recurrent units and have the strong similarity to recent successful elements applied in RNNs. Thanks to this technique as they were able to train the NN with 152 layers while still having lower complexity than the VGGNet. It achieves the top-5 error rate of 3.57%, which beats human-level performance on this dataset.



Q5. What is HAAR CASCADE?

Answer:

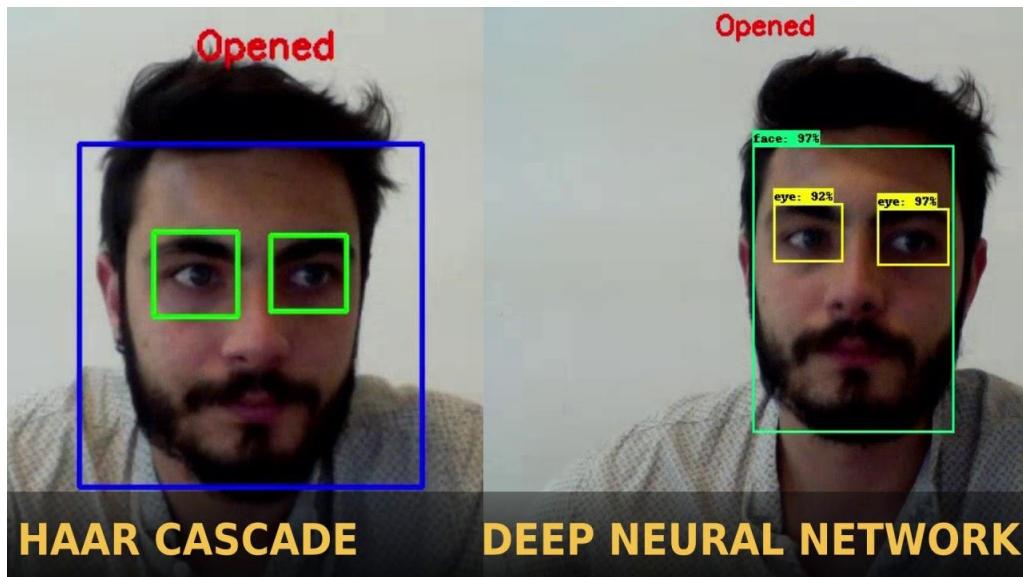
Haar Cascade: It is the machine learning object detections algorithm used to identify the objects in an image or the video and based on the concept of features proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001.

It is a machine learning-based approach where the cascade function is trained from the lot of positive and negative images. It is then used to detect the objects in other images.

The algorithm has four stages:

- Haar Feature Selection
- Creating Integral Images
- Adaboost Training
- Cascading Classifiers

It is well known for being able to detect faces and body parts in an image but can be trained to identify almost any object.

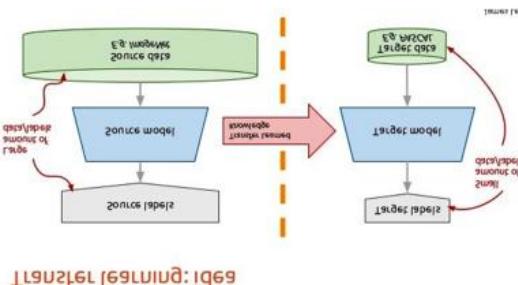


Q6. What is Transfer Learning?

Answer:

Transfer learning: It is the machine learning method where the model developed for a task is reused as the starting point for the model on the second task .

Transfer Learning differs from the traditional Machine Learning in that it is the use of pre-trained models that have been used for another task to jump-start the development process on a new task or problem.



The benefits of the Transfer Learning are that it can speed up the time as it takes to develop and train the model by reusing these pieces or modules of already developed models. This helps to speed up the model training process and accelerate results.

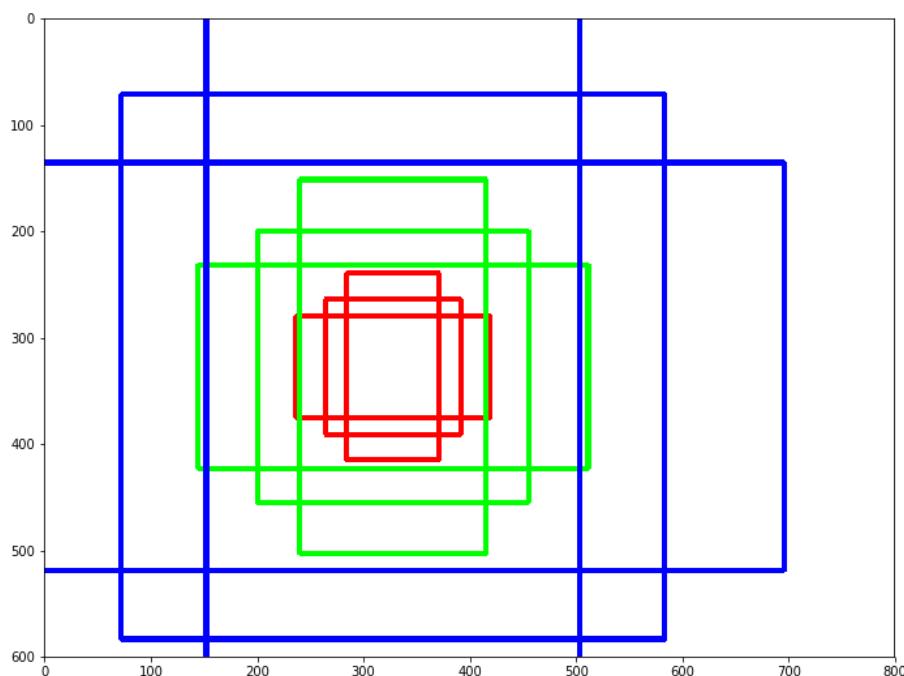
Q7. What is Faster, R-CNN?

Answer:

Faster R-CNN: It has two networks: region proposal network (RPN) for generating region proposals and a network using these proposals to detect objects. The main difference here with the Fast R-CNN is that the later uses selective search to generate the region proposals. The time cost of generating the region proposals is much smaller in the RPN than selective search, when RPN shares the most computation with object detection network. In brief, RPN ranks region boxes (called anchors) and proposes the ones most likely containing objects.

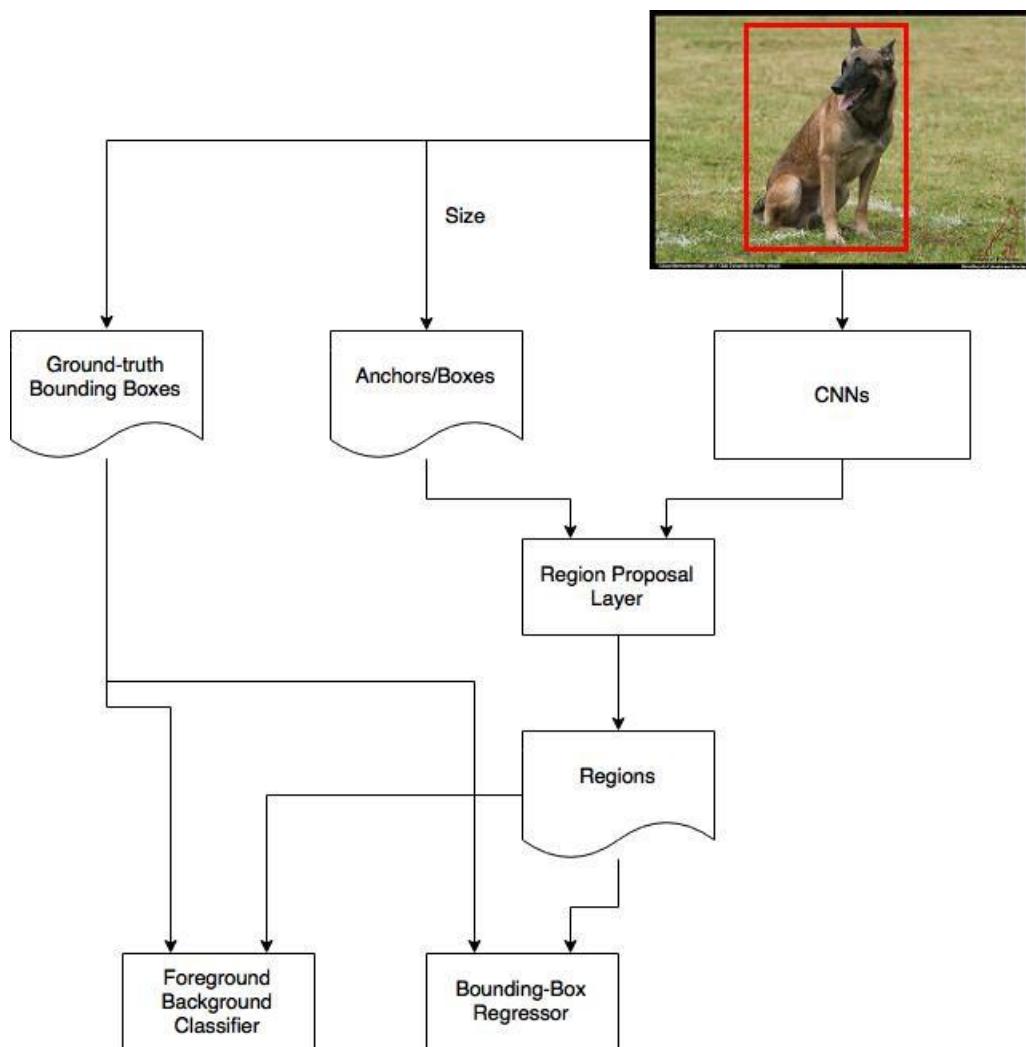
Anchors

Anchors play a very important role in Faster R-CNN. An anchor is the box. In default configuration of Faster R-CNN, there are nine anchors at the position of an image. The graph shows 9 anchors at the position (320, 320) of an image with size (600, 800).



Region Proposal Network:

The output of the region proposal network is the bunch of boxes/proposals that will be examined by a classifier and regressor to check the occurrence of objects eventually. To be more precise, RPN predicts the possibility of an anchor being background or foreground, and refine the anchor.

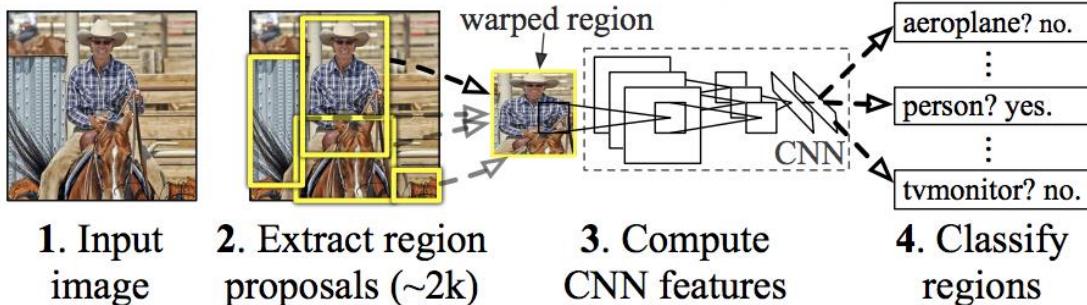


Q8. What is RCNN?

Answer:

To bypass the problem of selecting the huge number of regions, Ross Girshick et al. proposed a method where we use the selective search to extract just 2000 regions from the image, and he called them as region proposals. Therefore, instead of trying to classify the huge number of regions, you can work with 2000 regions.

R-CNN: Regions with CNN features



Problems with R-CNN:

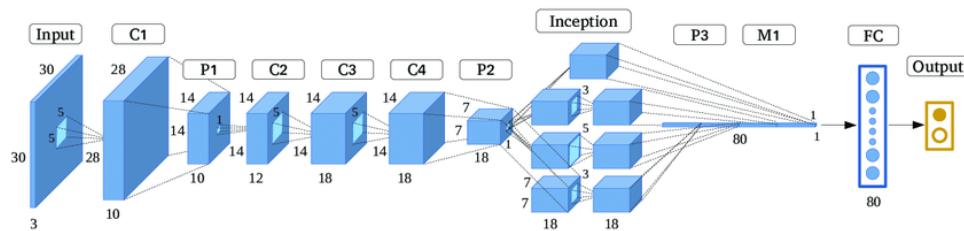
- It still takes the huge amount of time to train the network as we would have to classify 2000 region proposals per image.
- It cannot be implemented real-time as it takes around 47 seconds for each test image.
- The selective search algorithm is the fixed algorithm. Therefore, no learning is happening at that stage. This leads to the generation of the bad candidate region proposals.

Q9.What is GoogLeNet/Inception?

Answer:

The winner of the ILSVRC 2014 competition was GoogLeNet from Google. It achieved a top-5 error rate of 6.67%! This was very close to human-level performance which the organisers of the challenge were now forced to evaluate. As it turns out, this was rather hard to do and required some human training to beat GoogLeNets accuracy. After the few days of training, the human expert (Andrej Karpathy) was able to achieve the top-5 error rate of 5.1%(single model) and 3.6%(ensemble). The network used the CNN inspired by LeNet but implemented a novel element which is dubbed an inception module. It used batch normalisation, image distortions and RMSprop. This module is based on the several very small convolutions to reduce the number of parameters drastically. Their architecture consisted of the 22 layer deep CNN but reduced the number of parameters from 60 million (AlexNet) to 4 million.

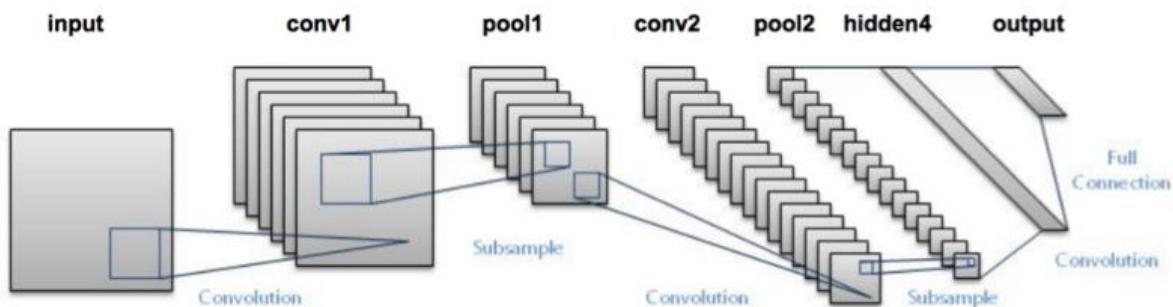
It contains 1×1 Convolution at the middle of network, and global average pooling is used at the end of the network instead of using the fully connected layers. These two techniques are from another paper “Network In-Network” (NIN). Another technique, called inception module, is to have different sizes/types of convolutions for the same input and to stack all the outputs.



Q10. What is LeNet-5?

Answer:

LeNet-5, a pioneering 7-level convolutional network by the LeCun et al in 1998, that classifies digits, was applied by several banks to recognise hand-written numbers on checks (cheques) digitised in 32x32 pixel greyscale input images. The ability to process higher-resolution images requires larger and more convolutional layers, so the availability of computing resources constrains this technique.

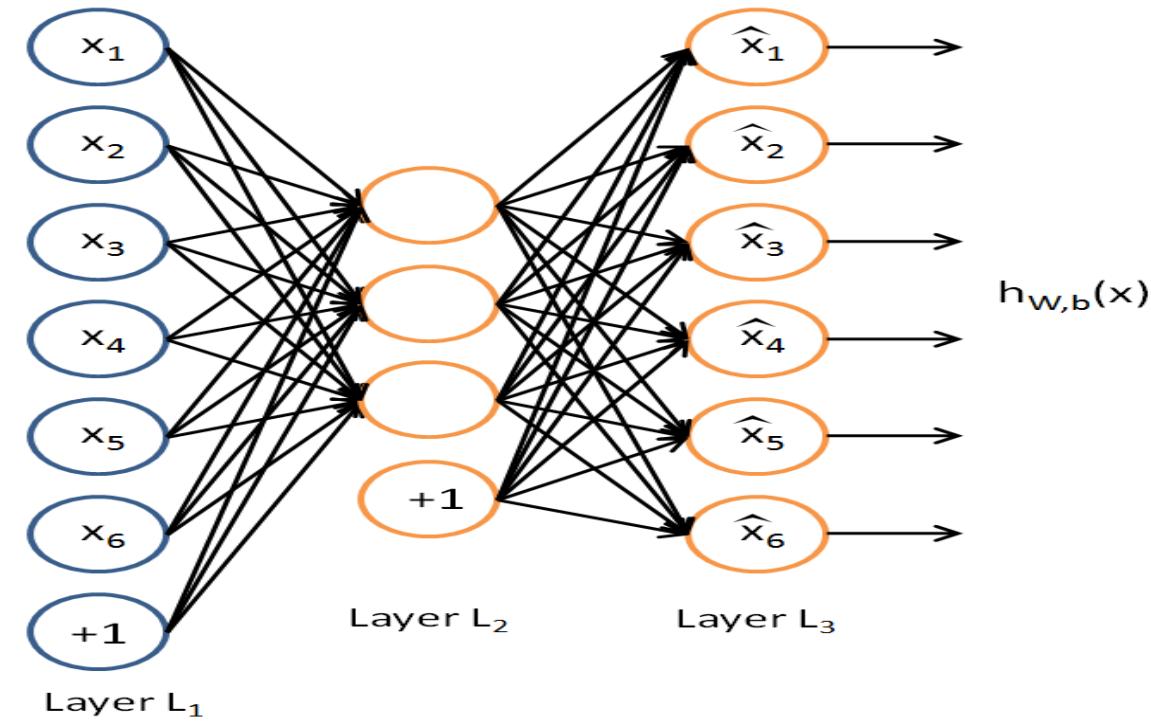


LeNet-5 is a very simple network. It only has seven layers, among which there are three convolutional layers (C1, C3 and C5), two sub-sampling (pooling) layers (S2 and S4), and one fully connected layer (F6), that are followed by output layers. Convolutional layers use 5 by 5 convolutions with stride 1. Sub-sampling layers are 2 by 2 average pooling layers. Tanh sigmoid activations are used throughout the network. Several interesting architectural choices were made in LeNet-5 that are not very common in the modern era of deep learning.

Q1. What is Autoencoder?

Answer:

Autoencoder neural network: It is an unsupervised Machine learning algorithm that applies backpropagation, setting the target values to be equal to the inputs. It is trained to attempt to copy its input to its output. Internally, it has the hidden layer that describes a code used to represent the input.



It is trying to learn the approximation to the identity function, to output \hat{x} that is similar to the x .

Autoencoders belongs to the neural network family, but they are also closely related to PCA (principal components analysis).

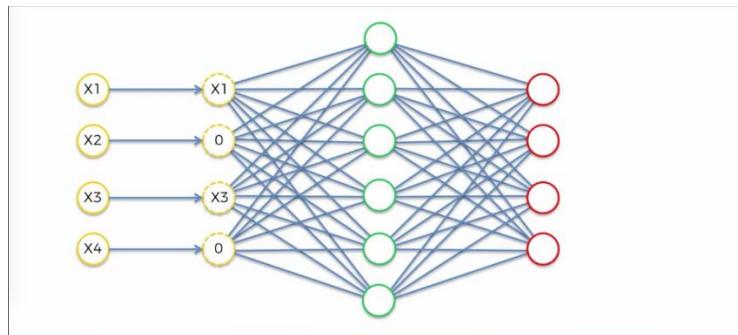
Auto encoders, although it is quite similar to PCA, but its Autoencoders are much more flexible than PCA. Autoencoders can represent both liners and non-linear transformation in encoding, but PCA can perform linear transformation. Autoencoders can be layered to form deep learning network due to its Network representation.

Types of Autoencoders:

1. Denoising autoencoder

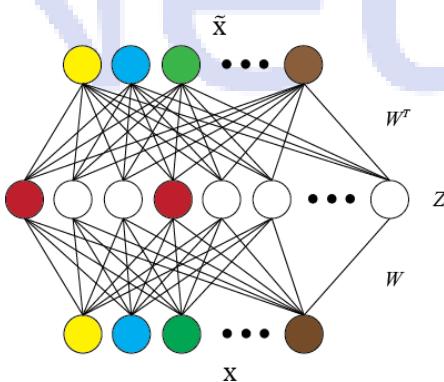
Autoencoders are Neural Networks which are used for feature selection and extraction. However, when there are more nodes in hidden layer than there are inputs, the Network is risking to learn so-called “Identity Function”, also called “Null Function”, meaning that output equals the input, marking the Autoencoder useless.

Denoising Autoencoders solve this problem by corrupting the data on purpose by randomly turning some of the input values to zero. In general, the percentage of input nodes which are being set to zero is about 50%. Other sources suggest a lower count, such as 30%. It depends on the amount of data and input nodes you have.



2. Sparse autoencoder

An autoencoder takes the input image or vector and learns code dictionary that changes the raw input from one representation to another. Where in sparse autoencoders with a sparsity enforcer that directs a single-layer network to learn code dictionary which in turn minimizes the error in reproducing the input while restricting number of code words for reconstruction. The sparse autoencoder consists a single hidden layer, which is connected to the input vector by a weight matrix forming the encoding step. The hidden layer then outputs to a reconstruction vector, using a tied weight matrix to form the decoder.



Q2. What Is Text Similarity?

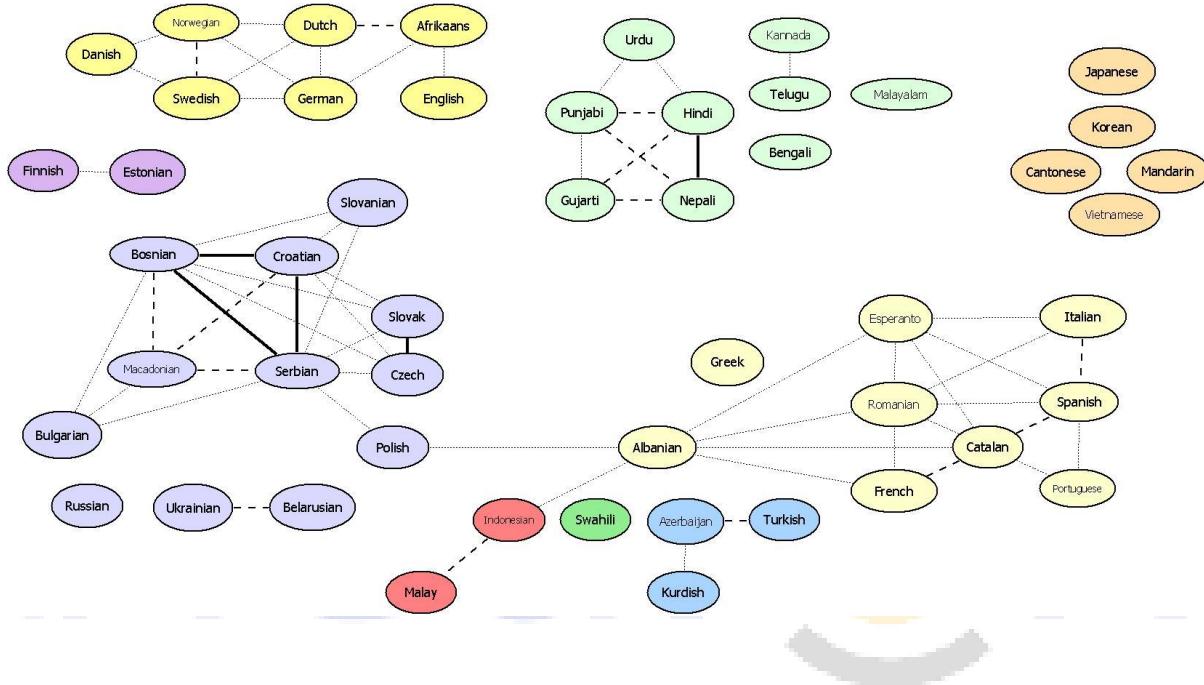
Answer:

When talking about text similarity, different people have a slightly different notion on what text similarity means. In essence, the goal is to compute how ‘close’ two pieces of text are in (1) meaning or (2) surface closeness. The first is referred to as **semantic similarity**, and the latter is referred to as **lexical similarity**. Although the methods for *lexical similarity* are often used to achieve *semantic similarity* (to a certain extent), achieving true semantic similarity is often much more involved.

Lexical or Word Level Similarity

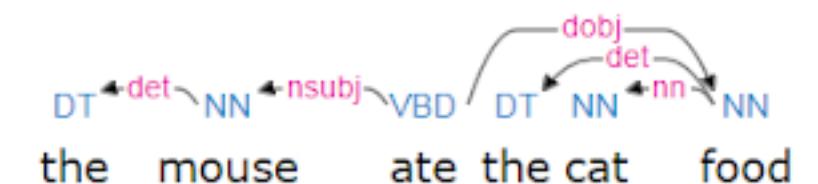
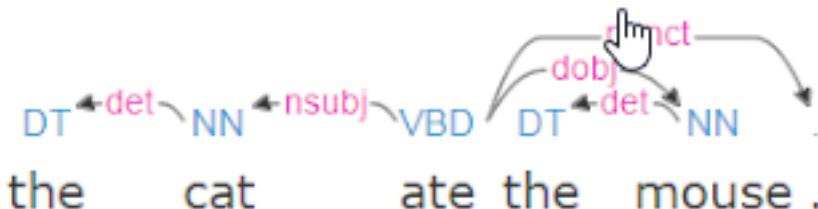
When referring to text similarity, people refer to how similar the two pieces of text are at the surface level. Example- how similar are the phrases “*the cat ate the mouse*” with “*the mouse ate the cat food*” by just looking at the words? On the surface, if you consider only word-level similarity, these two phrases (with determiners disregarded) appear very similar as 3 of the 4 unique words are an exact overlap.

$$\text{Overlap} = \text{'cat ate mouse'} \cap \text{'mouse ate cat food'} = 3$$



Semantic Similarity:

Another notion of similarity mostly explored by NLP research community is how similar in meaning are any two phrases? If we look at the phrases, “*the cat ate the mouse*” and “*the mouse ate the cat food*”. As we know that while the words significantly overlaps, these two phrases have different meaning. Meaning out of the phrases is often the more difficult task as it requires deeper level of analysis. Example, we can actually look at the simple aspects like order of words: “*cat==>ate==>mouse*” and “*mouse==>ate==>cat food*”. Words overlap in this case, the order of the occurrence is different, and we can tell that, these two phrases have different meaning. This is just the one example. Most people use the syntactic parsing to help with the semantic similarity. Let’s have a look at the parse trees for these two phrases. What can you get from it?



Q3. What is dropout in neural networks?

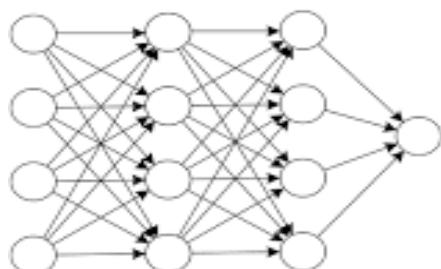
Answer:

When we training our neural network (or model) by updating each of its weights, it might become too dependent on the dataset we are using. Therefore, when this model has to make a prediction or classification, it will not give satisfactory results. This is known as over-fitting. We might understand this problem through a real-world example: If a student of science learns *only* one chapter of a book and then takes a test on the *whole* syllabus, he will probably fail.

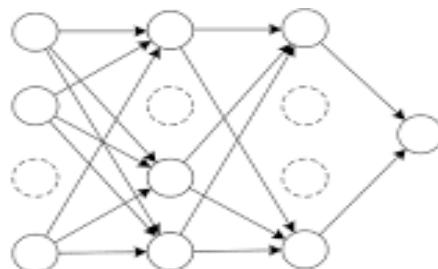
To overcome this problem, we use a technique that was introduced by Geoffrey Hinton in 2012. This technique is known as **dropout**.

Dropout refers to ignoring units (i.e., neurons) during the training phase of certain set of neurons, which is chosen at random. By “ignoring”, I mean these units are not considered during a particular forward or backward pass.

At each training stage, individual nodes are either dropped out of the net with probability $1-p$ or kept with probability p , so that a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed.



(a) Standard Neural Network

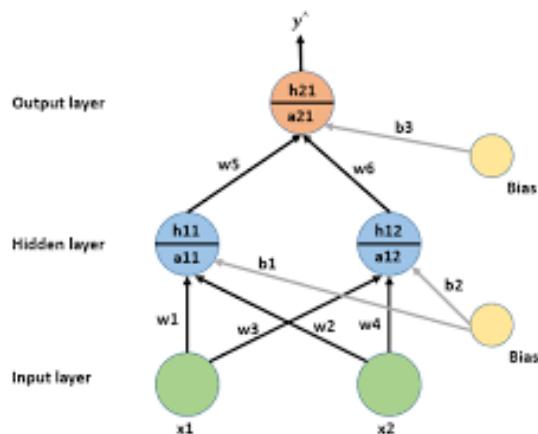


(b) Network after Dropout

Q4. What is Forward Propagation?

Answer:

Input X provides the information that then propagates to hidden units at each layer and then finally produce the output y . The architecture of network entails determining its depth, width, and the activation functions used on each layer. **Depth** is the number of the hidden layers. **Width** is the number of units (nodes) on each hidden layer since we don't control neither input layer nor output layer dimensions. There are quite a few set of activation functions such *Rectified Linear Unit*, *Sigmoid*, *Hyperbolic tangent*, etc. Research has proven that deeper networks outperform networks with more hidden units. Therefore, it's always better and won't hurt to train a deeper network.



Q5. What is Text Mining?

Answer:

Text mining: It is also referred as *text data mining*, roughly equivalent to *text analytics*, is the process of deriving high-quality information from text. High-quality information is typically derived through the devising of patterns and trends through means such as statistical pattern learning. Text mining usually involves the process of structuring the input text (usually parsing, along with the addition of some derived linguistic features and the removal of others, and subsequent insertion into a database), deriving patterns within the structured data, and finally evaluation and interpretation of the output. 'High quality' in text mining usually refers to some combination of relevance, novelty, and interest. Typical text mining tasks include text categorization, text clustering, concept/entity extraction, production of granular taxonomies, sentiment analysis, document summarization, and entity relation modeling (*i.e.*, learning relations between named entities).

Sources of Data

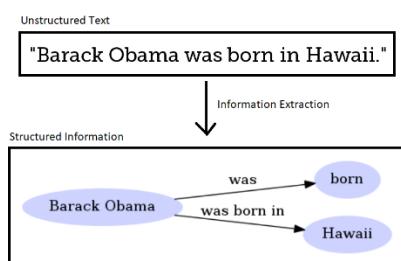


Q6. What is Information Extraction?

Answer:

Information extraction (IE): It is the task of automatically extracting structured information from the unstructured and/or semi-structured machine-readable documents. In most of the cases, this activity concerns processing human language texts using natural language processing (NLP).

Information extraction depends on named entity recognition (NER), a sub-tool used to find targeted information to extract. NER recognizes entities first as one of several categories, such as location (LOC), persons (PER), or organizations (ORG). Once the information category is recognized, an information extraction utility extracts the named entity's related information and constructs a machine-readable document from it, which algorithms can further process to extract meaning. IE finds meaning by way of other subtasks, including co-reference resolution, relationship extraction, language, and vocabulary analysis, and sometimes audio extraction.



Q7. What is Text Generation?

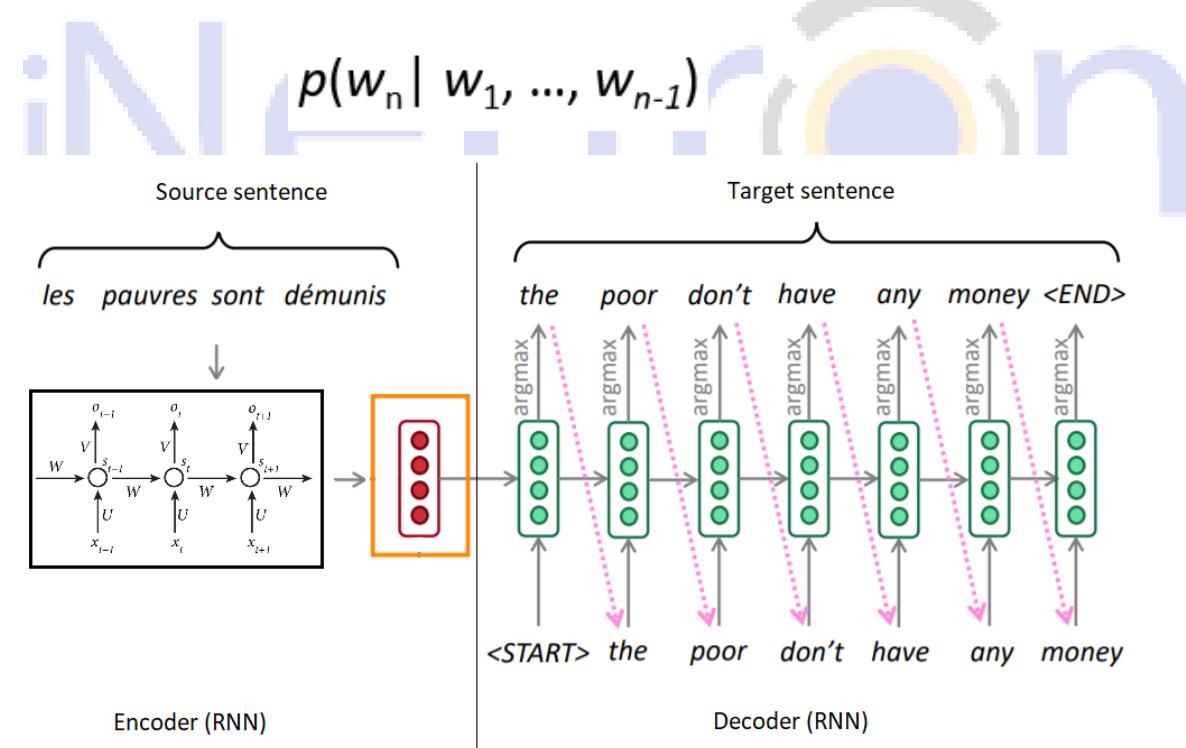
Answer:

Text Generation: It is a type of the Language Modelling problem. **Language Modelling** is the core problem for several of natural language processing tasks such as speech to text, conversational system, and the text summarization. The trained language model learns the likelihood of occurrence of the word based on the previous sequence of words used in the text. Language models can be operated at the character level, n-gram level, sentence level or even paragraph level.

A language model is at the core of many NLP tasks, and is simply a probability distribution over a sequence of words:

$$p(w_1, \dots, w_n)$$

It can also be used to estimate the conditional probability of the next word in a sequence:



Q8. What is Text Summarization?

Answer:

We all interact with the applications which uses the text summarization. Many of the applications are for the platform which publishes articles on the daily news, entertainment, sports. With our busy schedule, we like to read the summary of those articles before we decide to jump in for reading entire article. Reading a summary helps us to identify the interest area, gives a brief context of the story.

Text summarization is a subdomain of Natural Language Processing (NLP) that deals with extracting summaries from huge chunks of texts. There are two main types of techniques used for text summarization: NLP-based techniques and deep learning-based techniques.

Text summarization: It refers to the technique of shortening long pieces of text. The intention is to create the coherent and fluent summary having only the main points outlined in the document.

How text summarization works:

The two types of summarization, abstractive and the extractive summarization.

1. **Abstractive Summarization:** It select words based on the semantic understanding; even those words did not appear in the source documents. It aims at producing important material in the new way. They interprets and examines the text using advanced natural language techniques to generate the new shorter text that conveys the most critical information from the original text.

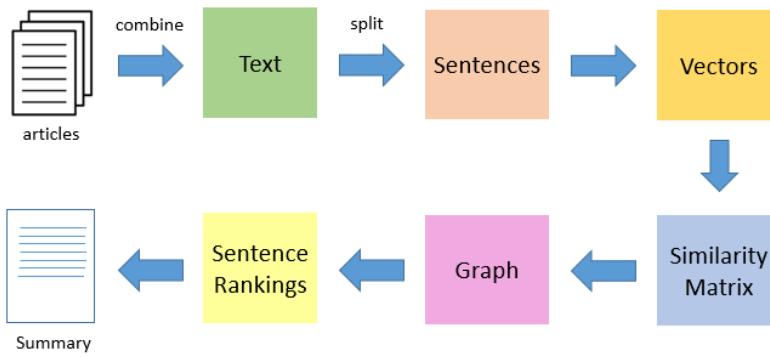
It can be correlated in the way human reads the text article or blog post and then summarizes in their word.

Input document → understand context → semantics → create own summary.

2. **Extractive Summarization:** It attempt to summarize articles by selecting the subset of words that retain the most important points.

This approach weights the most important part of sentences and uses the same to form the summary. Different algorithm and the techniques are used to define the weights for the sentences and further rank them based on importance and similarity among each other.

Input document → sentences similarity → weight sentences → select sentences with higher rank.



Q9. What is Topic Modelling?

Answer:

Topic Modelling is the task of using unsupervised learning to extract the main topics (represented as a set of words) that occur in a collection of documents.

Topic modeling, in the context of Natural Language Processing, is described as a method of uncovering hidden structure in a collection of texts.

Dimensionality Reduction:

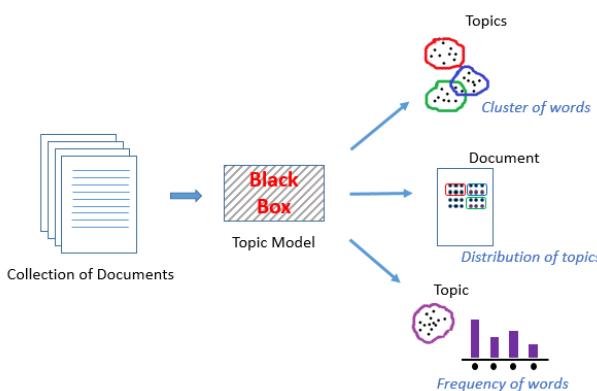
Topic modeling is the form of dimensionality reduction. Rather than representing the text T in its feature space as $\{\text{Word}_i: \text{count}(\text{Word}_i, T) \text{ for Word}_i \in V\}$, we can represent the text in its topic space as $(\text{Topic}_i: \text{weight}(\text{Topic}_i, T) \text{ for Topic}_i \in \text{Topics})$.

Unsupervised learning:

Topic modeling can be compared to the clustering. As in the case of clustering, the number of topics, like the number of clusters, is the hyperparameter. By doing the topic modeling, we build clusters of words rather than clusters of texts. A text is thus a mixture of all the topics, each having a certain weight.

A Form of Tagging

If document classification is assigning a single category to a text, topic modeling is assigning multiple tags to a text. A human expert can label the resulting topics with human-readable labels and use different heuristics to convert the weighted topics to a set of tags.

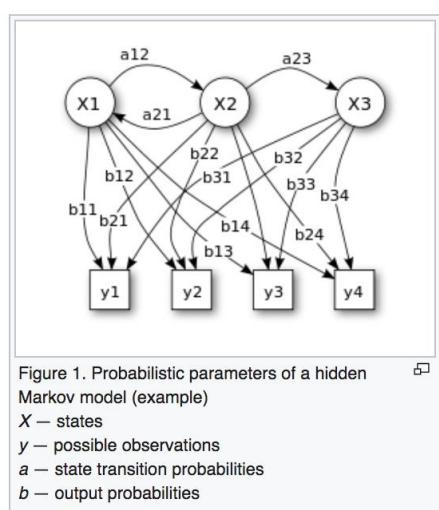


Q10.What is Hidden Markov Models?

Answer:

Hidden Markov Models (HMMs) are the class of probabilistic graphical model that allow us to predict the sequence of unknown (hidden) variables from the set of observed variables. The simple example of an HMM is predicting the weather (hidden variable) based on the type of clothes that someone wears (observed). An HMM can be viewed as the Bayes Net unrolled through time with observations made at the sequence of time steps being used to predict the best sequence of the hidden states.

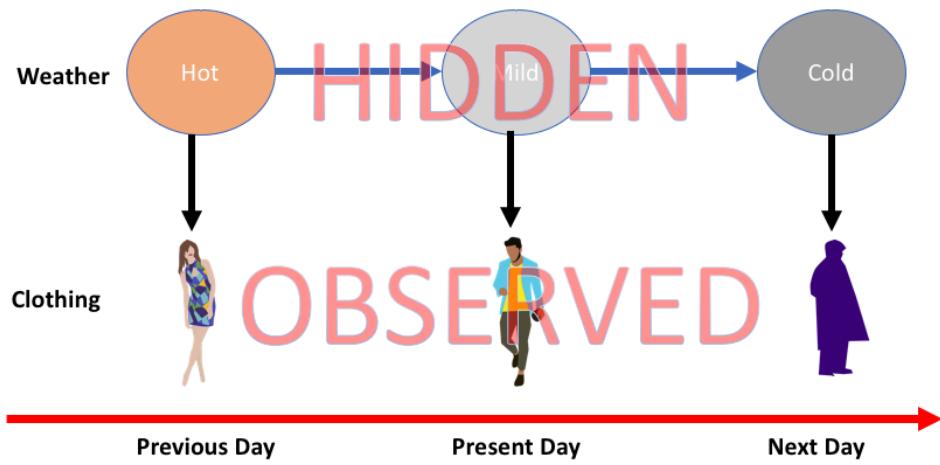
The below diagram from Wikipedia shows that HMM and its transitions. The scenario is the room that contains urns X1, X2, and X3, each of which contains a known mix of balls, each ball labeled y1, y2, y3, and y4. The sequence of four balls is randomly drawn. In this particular case, the user observes the sequence of balls y1,y2,y3, and y4 and is attempting to discern the hidden state, which is the right sequence of three urns that these four balls were pulled from.



Why Hidden, Markov Model?

The reason it is called the Hidden Markov Model is because we are constructing an inference model based on the assumptions of a Markov process. The Markov process assumption is simply that the “future is independent of the past given the present”.

To make this point clear, let us consider the scenario below where the weather, the hidden variable, can be hot, mild or cold, and the observed variables are the type of clothing worn. The arrows represent transitions from a hidden state to another hidden state or from a hidden state to an observed variable.



iNeuron

Q1.What is Statistics Learning?

Answer:

Statistical learning: It is the framework for understanding data based on the statistics, which can be classified as the supervised or unsupervised. Supervised statistical learning involves building the statistical model for predicting, or estimating, an output based on one or more inputs, while in unsupervised statistical learning, there are inputs but no supervising output, but we can learn relationships and structure from such data.

$$Y = f(X) + \epsilon, X = (X_1, X_2, \dots, X_p),$$

f : It is an unknown function & ϵ is random error (reducible & irreducible).

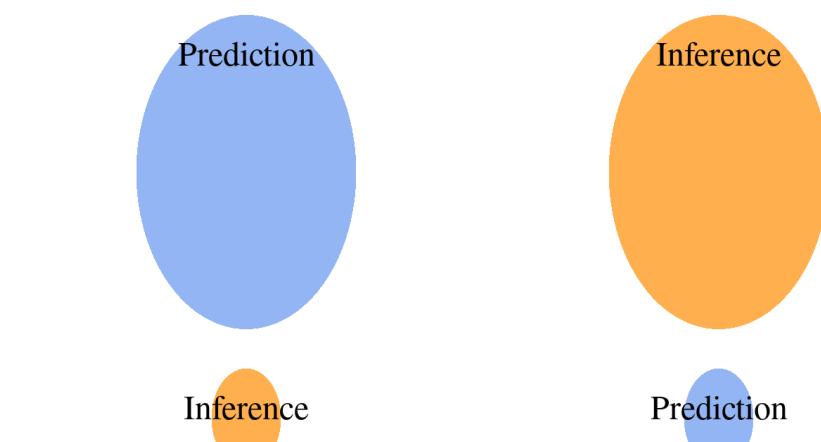
Prediction & Inference:

In the situations , where the set of inputs X are readily available, but the output Y is not known, we often treat f as the black box (not concerned with the exact form of “f”), as long as it yields the accurate predictions for Y. This is the *prediction*.

There are the situations where we are interested in understanding the way that Y is affected as X change. In this type of situation, we wish to estimate f , but our goal is not necessarily to make the predictions for Y . Here we are more interested in understanding the relationship between the X and Y . Now f cannot be treated as the black box, because we need to know its exact form. This is *inference*.

Machine Learning

Statistics

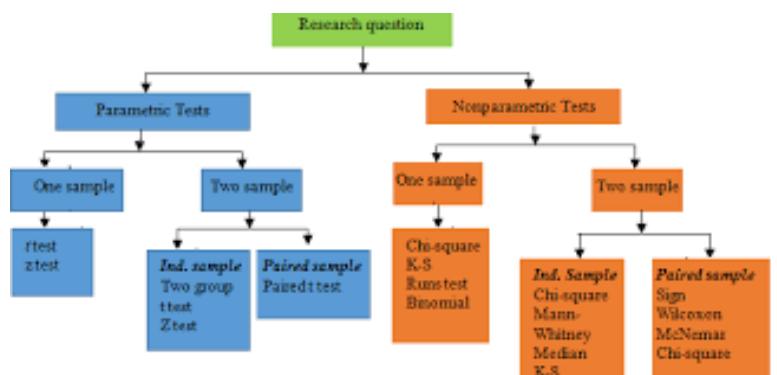


Parametric & Non-parametric methods

Parametric statistics: This statistical tests based on underlying the assumptions about data's distribution. In other words, It is based on the parameters of the normal curve. Because parametric statistics are based on the normal curve, data must meet certain assumptions, or parametric statistics cannot be calculated. Before running any parametric statistics, you should always be sure to test the assumptions for the tests that you are planning to run.

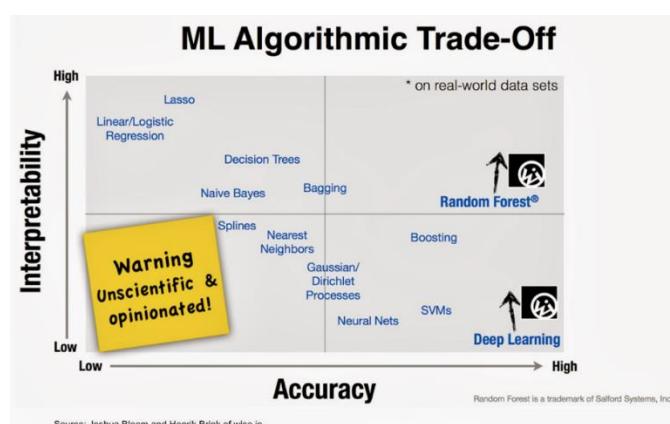
$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p$$

As by the name, nonparametric statistics are not based on parameters of the normal curve. Therefore, if our data violate the assumptions of a usual parametric and nonparametric statistics might better define the data, try running the nonparametric equivalent of the parametric test. We should also consider using nonparametric equivalent tests when we have limited sample sizes (e.g., $n < 30$). Though the nonparametric statistical tests have more flexibility than do parametric statistical tests, nonparametric tests are not as robust; therefore, most statisticians recommend that when appropriate, parametric statistics are preferred.



Prediction Accuracy and Model Interpretability:

Out of many methods that we use for the statistical learning, some are less flexible and more restrictive . When inference is the goal, then there are clear advantages of using the simple and relatively inflexible statistical learning methods. When we are only interested in the prediction, we use flexible models available.



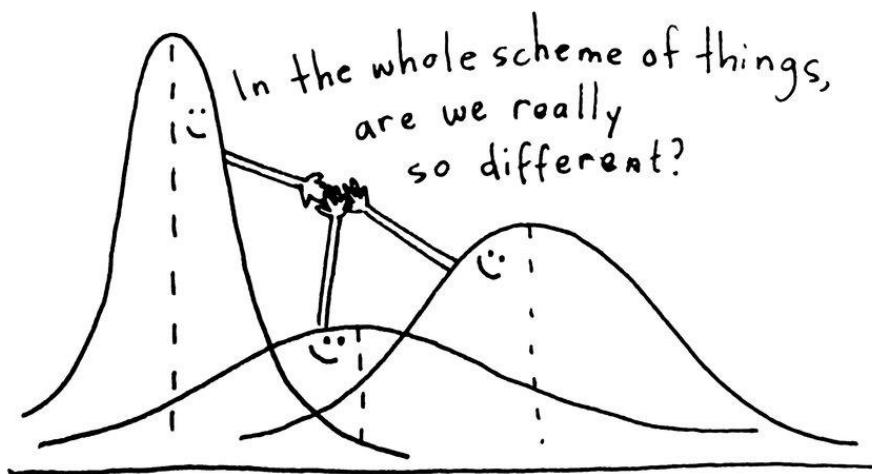
Q2. What is ANOVA?

Answer:

ANOVA: it stands for “ Analysis of Variance ” is an extremely important tool for analysis of data (both One Way and Two Way ANOVA is used). It is a statistical method to compare the population means of two or more groups by analyzing variance. The variance would differ only when the means are significantly different.

ANOVA test is the way to find out if survey or experiment results are significant. In other words, It helps us to figure out if we need to reject the null hypothesis or accept the alternate hypothesis. We are testing groups to see if there's a difference between them. Examples of when we might want to test different groups:

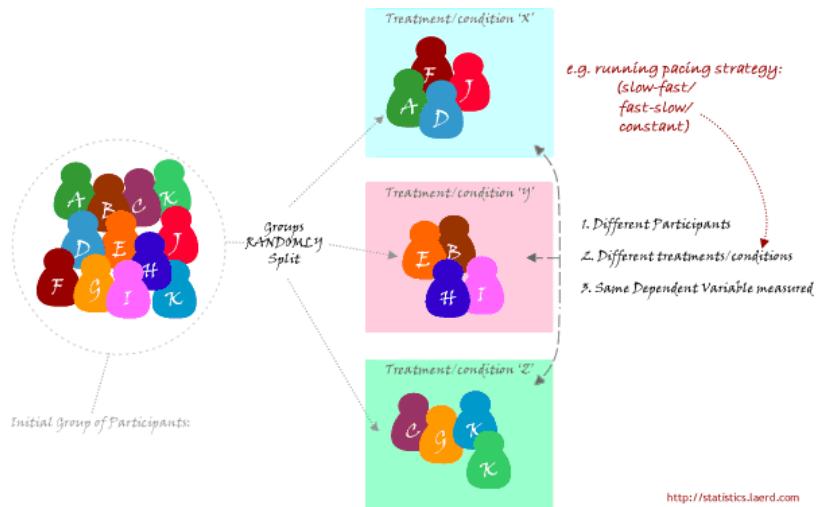
- The group of psychiatric patients are trying three different therapies: counseling, medication, and biofeedback. We want to see if one therapy is better than the others.
- The manufacturer has two different processes to make light bulbs if they want to know which one is better.
- Students from the different colleges take the same exam. We want to see if one college outperforms the other.



Types of ANOVA:

- One-way ANOVA
- Two-way ANOVA

One-way ANOVA is the hypothesis test in which only one categorical variable or the single factor is taken into consideration. With the help of F-distribution, it enables us to compare means of three or more samples. The Null hypothesis (H_0) is the equity in all population means while an Alternative hypothesis is the difference in at least one mean.



There are two-ways ANOVA examines the effect of two independent factors on a dependent variable. It also studies the inter-relationship between independent variables influencing the values of the dependent variable, if any.



Q3. What is ANCOVA?

Answer:

Analysis of Covariance (ANCOVA): It is the inclusion of the continuous variable in addition to the variables of interest (the dependent and independent variable) as means for the control. Because the ANCOVA is the extension of the ANOVA, the researcher can still assess main effects and the interactions to answer their research hypotheses. The difference between ANCOVA and an ANOVA is that an ANCOVA model includes the “covariate” that is correlated with dependent variable and means on dependent variable are adjusted due to effects the covariate has on it. Covariates can also

be used in many ANOVA based designs: such as between-subjects, within-subjects (repeated measures), mixed (between – and within – designs), etc. Thus, this technique answers the question

In simple terms, The difference between ANOVA and the ANCOVA is the letter "C", which stands for 'covariance'. Like ANOVA, "Analysis of Covariance" (ANCOVA) has the single continuous response variable. Unlike ANOVA, ANCOVA compares the response variable by both the factor and a continuous independent variable (example comparing test score by both 'level of education' and the 'number of hours spent in studying'). The terms for the continuous independent variable (IV) used in the ANCOVA is "covariate".

Example of ANCOVA

ANCOVA EXAMPLE

Independent Variables

(Factor)

Level of Education
(High School, College Degree,
or Graduate Degree)

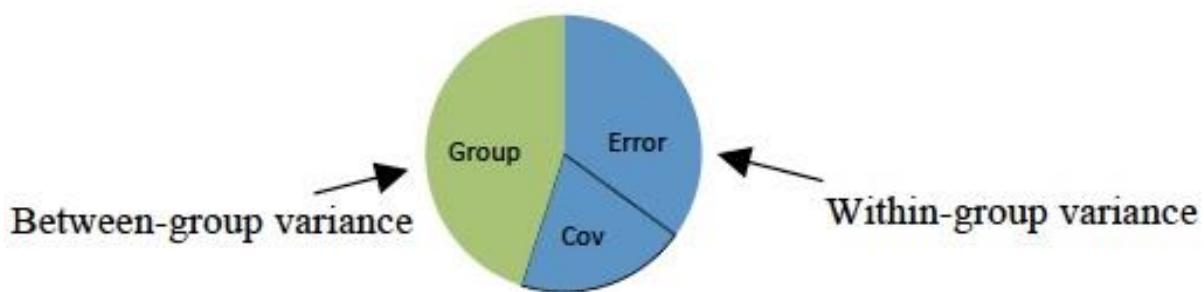
(Covariate)

Number of Hours
Spent Studying

Dependent Variable

(Response)

Test Score



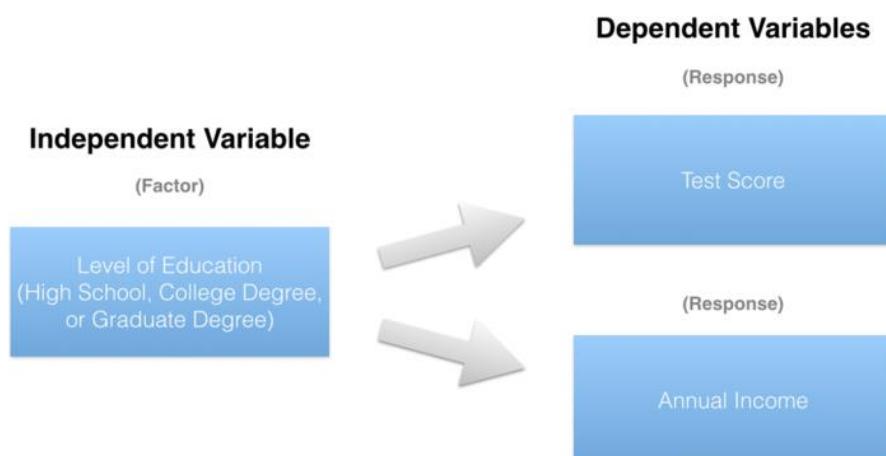
Q4. What is MANOVA?

Answer:

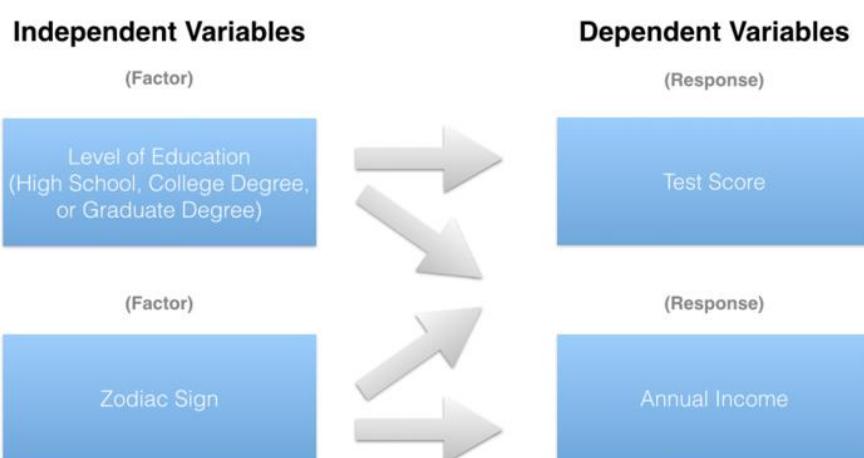
MANOVA (multivariate analysis of variance): It is a type of multivariate analysis used to analyze data that involves more than one dependent variable at a time. MANOVA allows us to test hypotheses regarding the effect of one or more independent variables on two or more dependent variables.

The obvious difference between ANOVA and the "Multivariate Analysis of Variance" (MANOVA) is the "M", which stands for multivariate. In basic terms, MANOVA is an ANOVA with two or more continuous response variables. Like ANOVA, MANOVA has both the one-way flavor and a two-way flavor. The number of factor variables involved distinguish the one-way MANOVA from a two-way MANOVA.

ONE-WAY MANOVA EXAMPLE



TWO-WAY MANOVA EXAMPLE



When comparing the two or more continuous response variables by the single factor, a one-way MANOVA is appropriate (e.g. comparing 'test score' and 'annual income' together by 'level of

education'). The two-way MANOVA also entails two or more continuous response variables, but compares them by at least two factors (e.g. comparing 'test score' and 'annual income' together by both 'level of education' and 'zodiac sign').

Q5. What is MANCOVA?

Answer:

Multivariate analysis of covariance (MANCOVA): It is a statistical technique that is the extension of analysis of covariance (ANCOVA). It is the multivariate analysis of variance (MANOVA) with a covariate(s)). In MANCOVA, we assess for statistical differences on multiple continuous dependent variables by an independent grouping variable, while controlling for a third variable called the covariate; multiple covariates can be used, depending on the sample size. Covariates are added so that it can reduce error terms and so that the analysis eliminates the covariates' effect on the relationship between the independent grouping variable and the continuous dependent variables.

ANOVA and ANCOVA, the main difference between the MANOVA and MANCOVA, is the "C," which again stands for the "covariance." Both the MANOVA and MANCOVA feature two or more response variables, but the key difference between the two is the nature of the IVs. While the MANOVA can include only factors, an analysis evolves from MANOVA to MANCOVA when one or more covariates are added to the mix.

MANCOVA EXAMPLE

Independent Variables

(Factor)
Level of Education
(High School, College Degree, or Graduate Degree)

(Covariate)
Number of Hours Spent Studying

Dependent Variables

(Response)
Test Score

(Response)
Annual Income



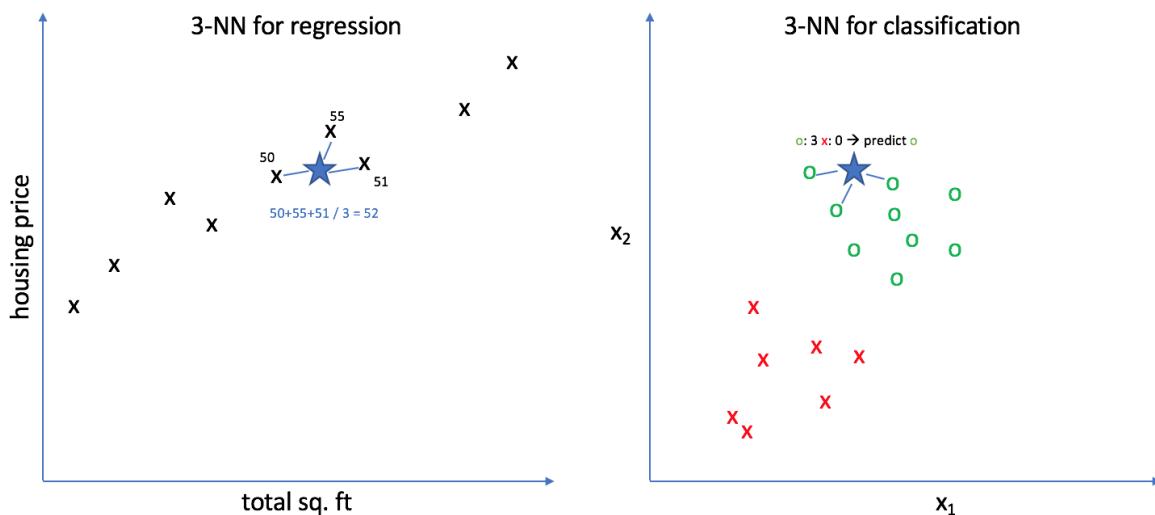
Q6. Explain the differences between KNN classifier and KNN regression methods.

Answer:

They are quite similar. Given a value for KK and a prediction point x_0 , KNN regression first identifies the KK training observations that are closest to x_0 , represented by N_0 . It then estimates $f(x_0)$ using the average of all the training responses in N_0 . In other words,

$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in N_0} y_i$$

So the main difference is the fact that for the classifier approach, the algorithm assumes the outcome as the class of more presence, and on the regression approach, the response is the average value of the nearest neighbors.



Q7. What is t-test?

Answer:

To understand T-Test Distribution, Consider the situation, you want to compare the performance of two workers of your company by checking the average sales done by each of them, or to compare the performance of a worker by comparing the average sales done by him with the standard value. In such situations of daily life, t distribution is applicable.

A t-test is the type of inferential statistic used to determine if there is a significant difference between the means of two groups, which may be related in certain features. It is mostly used when the data sets, like the data set recorded as the outcome from flipping a coin 100 times, would follow a normal distribution and may have unknown variances. A t-test is used as a hypothesis testing tool, which allows testing of an assumption applicable to a population.

Understand t-test with Example: Let's say you have a cold, and you try a naturopathic remedy. Your cold lasts a couple of days. The next time when you have a cold, you buy an over-the-counter pharmaceutical, and the cold lasts a week. You survey your friends, and they all tell you that their colds were of a shorter duration (an average of 3 days) when they took the homeopathic remedy. What you want to know is, are these results repeatable? A t-test can tell you by comparing the means of the two groups and letting you know the probability of those results happening by chance.

Type	T-statistic	Degrees of freedom
------	-------------	--------------------

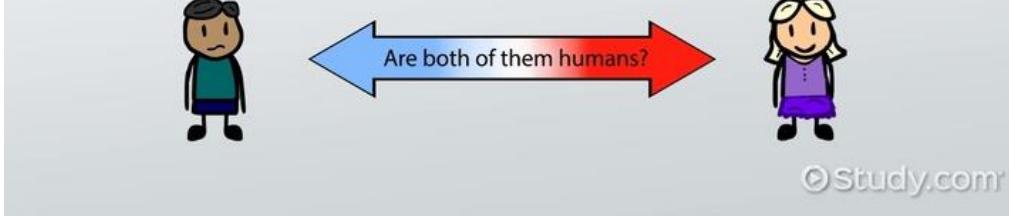
One-sample t-test $t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$ $df = n - 1$

Paired t-test $t = \frac{\bar{X}_D - \mu_0}{s_D/\sqrt{n}}.$ $df = n - 1$

OVERVIEW OF T-TESTS

T-test

Used to compare two samples to determine if they came from the same population.



Q8. What is Z-test?

Answer:

Z-test: It is a statistical test used to determine whether the two population means are different when the variances are known, and the sample size is large. The test statistic is assumed to have the normal distribution, and nuisance parameters such as standard deviation should be known for an accurate z-test to be performed.

Another definition of Z-test: A Z-test is a type of hypothesis test. Hypothesis testing is just the way for you to figure out if results from a test are valid or repeatable. Example, if someone said they had found the new drug that cures cancer, you would want to be sure it was probably true. Hypothesis test will tell you if it's probably true or probably not true. A Z test is used when your data is approximately normally distributed.

Z-Tests Working :

Tests that can be conducted as the z-tests include one-sample location test, a two-sample location test, a paired difference test, and a maximum likelihood estimate. Z-tests are related to t-tests, but t-tests are best performed when an experiment has the small sample size. Also, T-tests assumes the standard deviation is unknown, while z-tests assumes that it is known. If the standard deviation of the population is unknown, then the assumption of the sample variance equaling the population variance is made.

When we can run the Z-test :

Different types of tests are used in the statistics (i.e., f test, chi-square test, t-test). You would use a Z test if:

- Your sample size is greater than 30. Otherwise, use a t-test.
- Data points should be independent from each other. Some other words, one data point is not related or doesn't affect another data point.
- Your data should be normally distributed. However, for large sample sizes (over 30), this doesn't always matter.
- Your data should be randomly selected from a population, where each item has an equal chance of being selected.
- Sample sizes should be equal, if at all possible.

Z-TEST

Formula to find the value of Z (z-test) is:

$$Z = \frac{\bar{x} - \mu_0}{\sigma/\sqrt{n}}$$

 \bar{x} = mean of sample

 μ_0 = mean of population

 σ = standard deviation of population

 n = no. of observations

Q9. What is Chi-Square test?

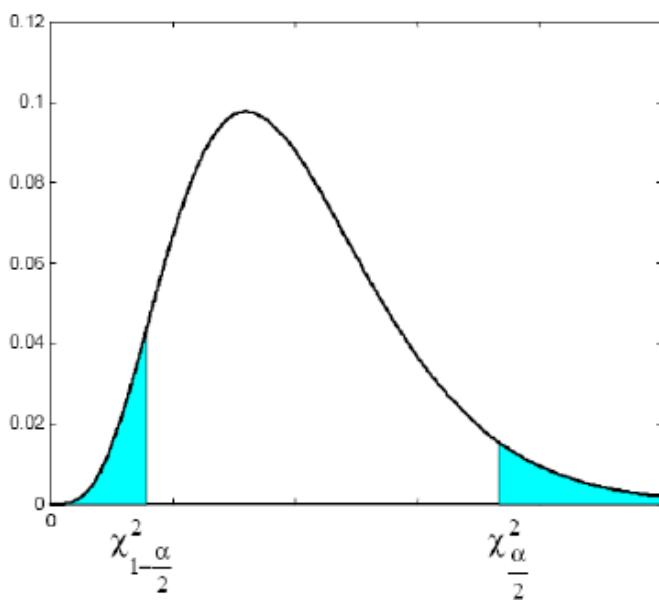
[Answer:](#)

Chi-square (χ^2) statistic: It is a test that measures how expectations compare to actual observed data (or model results). The data used in calculating a chi-square statistic must be random, raw, mutually exclusive, drawn from independent variables, and drawn from a large enough sample. For example, the results of tossing a coin 100 times meet these criteria.

Chi-square test is intended to test how it is that an observed distribution is due to chance. It is also called the "**goodness of fit**" statistic because it measures how well the observed distribution of the data fits with the distribution that is expected if the variables are independent.

Chi-square test is designed to analyze the **categorical** data. That means that the data has been counted and divided into categories. It will not work with parametric or continuous data (such as height in inches). For example, if you want to test whether attending class influences how students perform on an exam, using test scores (from 0-100) as data would not be appropriate for a Chi-square test. However, arranging students into the categories "Pass" and "Fail" would. Additionally, the data in a Chi-square grid should not be in the form of percentages, or anything other than frequency (count) data.

$$\chi^2_c = \sum \frac{(O_i - E_i)^2}{E_i}$$



Q10. What is correlation and the covariance in the statistics?

Answer:

The Covariance and Correlation are two mathematical concepts; these two approaches are widely used in the statistics. Both Correlation and the Covariance establish the relationship and also measures the dependency between the two random variables, the work is similar between these two, in the mathematical terms, they are different from each other.

Correlation: It is the statistical technique that can show whether and how strongly pairs of variables are related. For example, height and weight are related; taller people tend to be heavier than shorter people. The relationship isn't perfect. People of the same height vary in weight, and you can easily think of two people you know where the shorter one is heavier than the taller one. Nonetheless, the average weight of people 5'5" is less than the average weight of people 5'6", and their average weight is less than that of people 5'7", etc. Correlation can tell you just how much of the variation in peoples' weights is related to their heights.

Correlation Formula


$$\rho_{xy} = \frac{\text{Cov}(r_x, r_y)}{\sigma_x \sigma_y}$$
 

Covariance: It measures the directional relationship between the returns on two assets. The positive covariance means that asset returns move together while a negative covariance means they move inversely. Covariance is calculated by analyzing at-return surprises (standard deviations from the expected return) or by multiplying the correlation between the two variables by the standard deviation of each variable.



Covariance Formula

For Population

$$\text{Cov}(x,y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{N}$$

For Sample

$$\text{Cov}(x,y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{(N - 1)}$$

Q1. What is ERM (Empirical Risk Minimization)?

Answer:

Empirical risk minimization (ERM): It is a principle in statistical learning theory which defines a family of learning algorithms and is used to give theoretical bounds on their performance. The idea is that we don't know exactly how well an algorithm will work in practice (the true "risk") because we don't know the true distribution of data that the algorithm will work on, but as an alternative we can measure its performance on a known set of training data.

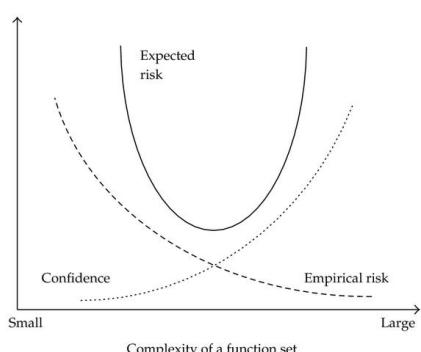
We assumed that our samples come from this distribution and use our dataset as an approximation. If we compute the loss using the data points in our dataset, it's called empirical risk. It is "empirical" and not "true" because we are using a dataset that's a subset of the whole population.

When our learning model is built, we have to pick a function that minimizes the empirical risk that is the delta between predicted output and actual output for data points in the dataset. This process of finding this function is called empirical risk minimization (ERM). We want to minimize the true risk. We don't have information that allows us to achieve that, so we hope that this empirical risk will almost be the same as the true empirical risk.

Let's get a better understanding by Example

We would want to build a model that can differentiate between a male and a female based on specific features. If we select 150 random people where women are really short, and men are really tall, then the model might incorrectly assume that height is the differentiating feature. For building a truly accurate model, we have to gather all the women and men in the world to extract differentiating features. Unfortunately, that is not possible! So we select a small number of people and hope that this sample is representative of the whole population.

$$\begin{aligned}
 R_{emp}[f] &= \sum_{x \in X} \sum_{j=1}^2 c(x, y_j, f(x)) p_{emp}(y_j, x) \\
 &= \frac{1}{n} \sum_{i=1}^n c(x_i, y_i, f(x_i))
 \end{aligned}$$



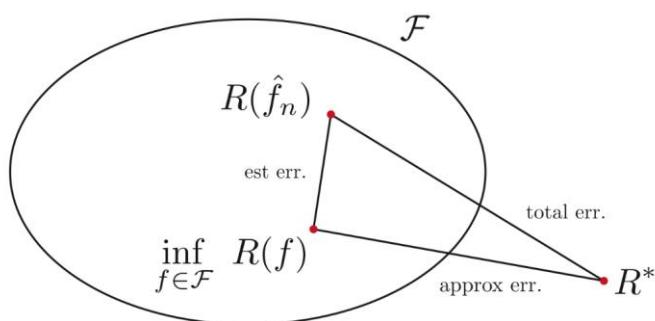
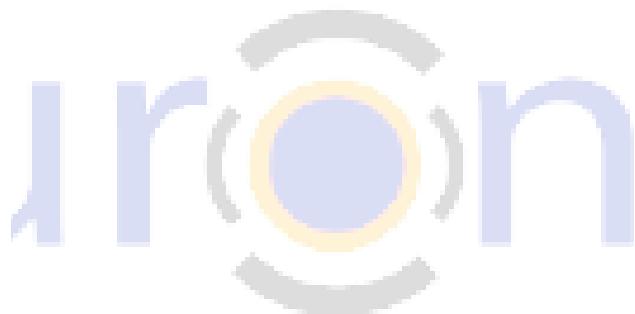
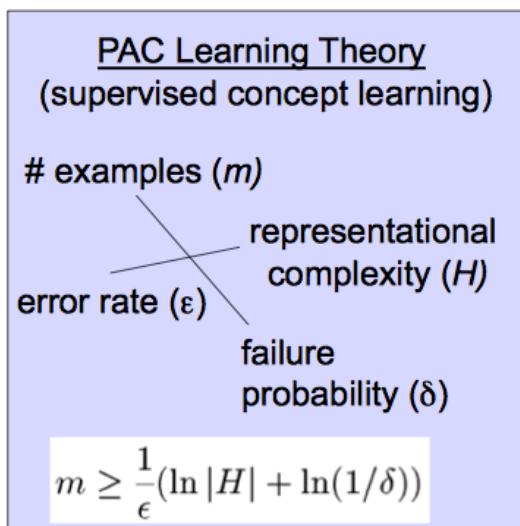
Q2. What is PAC (Probably Approximately Correct)?

Answer:

PAC: In computational learning theory, probably approximately correct (PAC) learning is a framework for mathematical analysis of machine learning.

The learner receives samples and must have to pick a generalization function (called the *hypothesis*) from a specific class of possible functions. Our goal is that, with high probability, the selected function will have low generalization error. The learner must be able to learn the concept given any arbitrary approximation ratio, probability of success, or distribution of the samples.

Hypothesis class is PAC(Probably Approximately Correct) learnable if there exists a function m_H and algorithm that for any labeling function f , distribution D over the domain of inputs X , **delta** and **epsilon** that with $m \geq m_H$ produces a hypothesis h like that with probability $1-\delta$ it returns a **true error** lower than **epsilon**. Labeling function is nothing other than saying that we have a specific function f that labels the data in the domain.



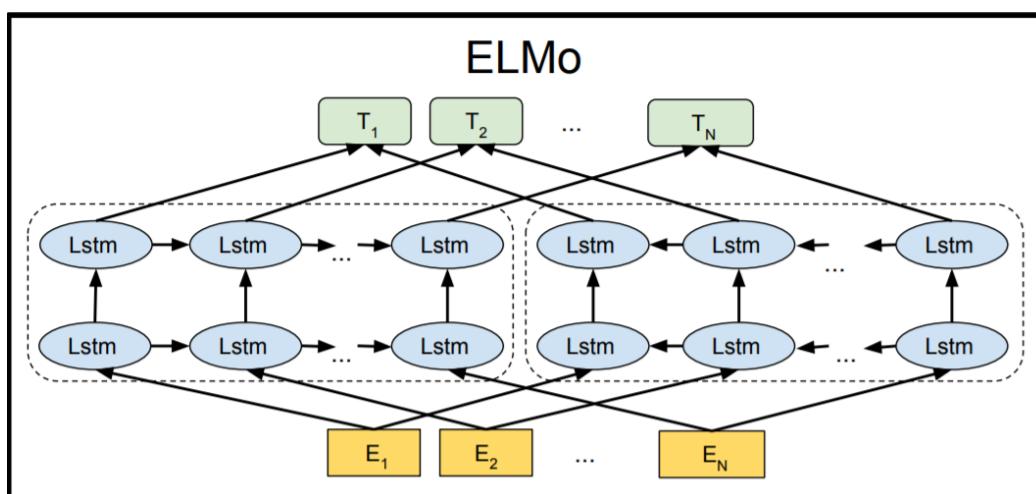
Q3. What is ELMo?

Answer:

ELMo is a novel way to represent words in vectors or embeddings. These word embeddings help achieve state-of-the-art (SOTA) results in several NLP tasks:

Task	Previous SOTA	ELMo + Baseline
SQuAD	SAN	84.4
SNLI	Chen et al (2017)	88.6
SRL	He et al (2017)	81.7
Coref	Lee et al (2017)	67.2
NER	Peters et al (2017)	91.93 +/- 0.19
Sentiment (5-class)	McCann et al (2017)	53.7
		54.7 +/- 0.5

It is a deep contextualized word representation that models both complex characteristics of word use (e.g., syntax and semantics), and how these uses vary across linguistic contexts. These word vectors are learned functions of internal states of a deep biLM(bidirectional language model), which is pre-trained on large text corpus. They could be easily added to existing models and significantly improve state of the art across a broad range of challenging NLP problems, including question answering, textual entailment and sentiment analysis.



Q4. What is Pragmatic Analysis in NLP?

Answer:

Pragmatic Analysis(PA): It deals with outside word knowledge, which means understanding i.e external to documents and queries. PA that focuses on what was described is reinterpreted by what it actually meant, deriving the various aspects of language that require real-world knowledge.

It deals with overall communicative and social content and its effect on interpretation. It means abstracting the meaningful use of language in situations. In this analysis, the main focus always on what was said in reinterpreted on what is intended.

It helps users to discover this intended effect by applying a set of rules that characterize cooperative dialogues.

E.g., "close the window?" should be interpreted as a request instead of an order.

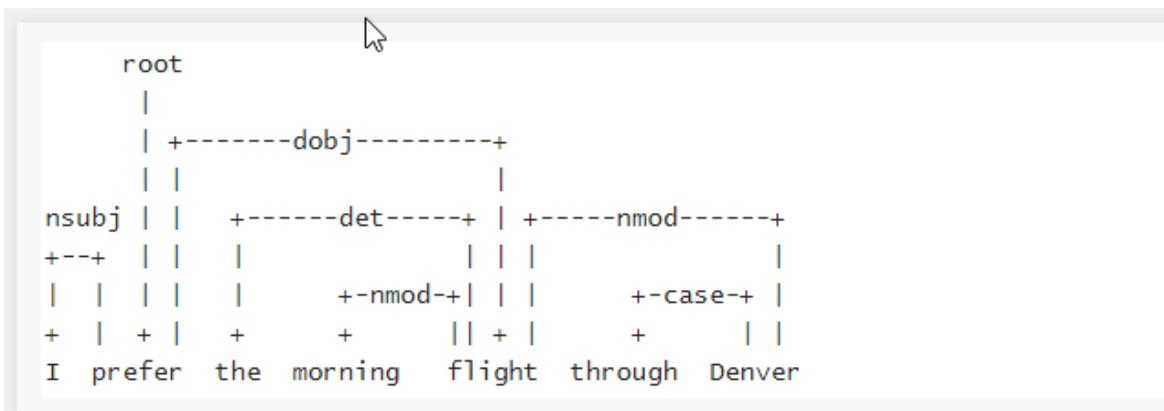
Personal	Organizational devices (Calls/Vocatives)
Request	Questions
Action	Answers/Responses
Permission	Repetition/Imitation
Offering/showing	Elicited identification
Descriptions	Routines
Statements (Internal reports)	Exclamations
Acknowledgements	Unclassified
Performatives	Double coded

Q5. What is Syntactic Parsing?

Answer:

Syntactic Parsing or Dependency Parsing: It is a task of recognizing a sentence and assigning a syntactic structure to it. Most Widely we used syntactic structure is the parse tree which can be generated using some parsing algorithms. These parse trees are useful in various applications like grammar checking or more importantly, it plays a critical role in the semantic analysis stage. For example to answer the question "*Who is the point guard for the LA Laker in the next game ?*" we need to figure out its subject, objects, attributes to help us figure out that the user wants the point guard of the LA Lakers specifically for the next game.

Example:



Q6. What is ULMFiT?

Answer:

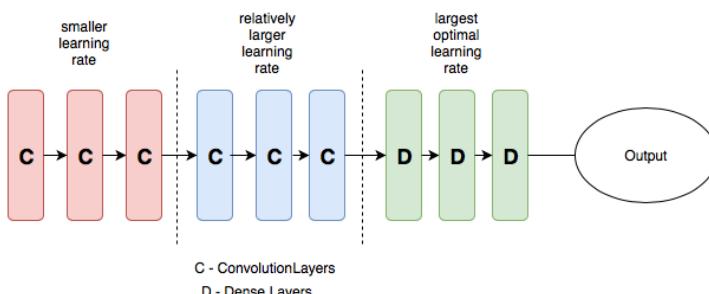
Transfer Learning in **NLP(Natural language Processing)** is an area that had not been explored with great success. But, in May 2018, **Jeremy Howard** and **Sebastian Ruder** came up with the paper – **Universal Language Model Fine-tuning for Text Classification(ULMFiT)** which explores the benefits of using a pre trained model on text classification. It proposes **ULMFiT(Universal Language Model Fine-tuning for Text Classification)**, a transfer learning method that could be applied to any task in NLP. In this method outperforms the state-of-the-art on six text classification tasks.

ULMFiT uses a **regular LSTM** which is the state-of-the-art language model architecture (**AWD-LSTM**). The LSTM network has three layers. Single architecture is used throughout – for pre-training as well as for fine-tuning.

ULMFiT achieves the state-of-the-art result using novel techniques like:

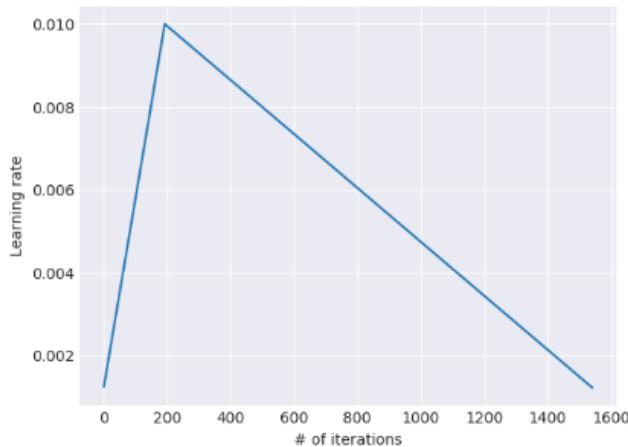
- Discriminative fine-tuning
- Slanted triangular learning rates
- Gradual unfreezing

Discriminative Fine-Tuning



Different layers of a neural network capture different types of information so they should be fine-tuned to varying extents. Instead of using the same learning rates for all layers of the model, discriminative fine-tuning allows us to tune each layer with different learning rates.

Slanted triangular learning



The model should quickly converge to a suitable region of the parameter space in the beginning of training and then later refine its parameters. Using a constant learning rate throughout training is not the best way to achieve this behaviour. Instead Slanted Triangular Learning Rates (STLR) linearly increases the learning rate at first and then linearly decays it.

Gradual Unfreezing

Gradual unfreezing is the concept of unfreezing the layers gradually, which avoids the catastrophic loss of knowledge possessed by the model. It first unfreezes the top layer and fine-tunes all the unfrozen layers for 1 epoch. It then unfreezes the next lower frozen layer and repeats until all the layers have been fine-tuned until convergence at the last iteration.

Q7. What is BERT?

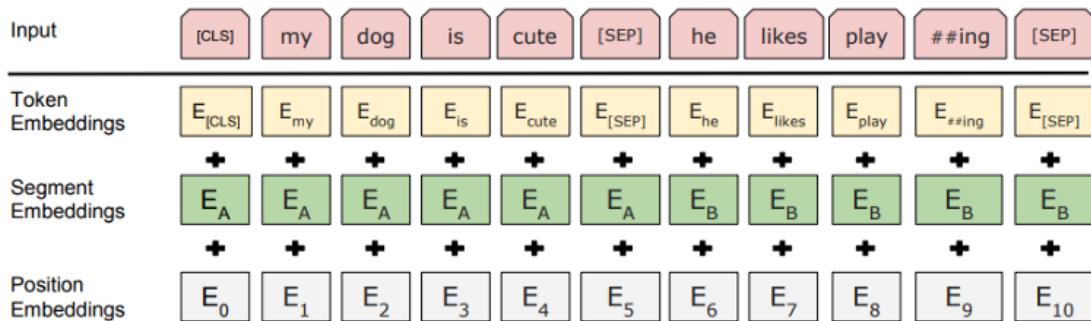
Answer:

BERT (Bidirectional Encoder Representations from Transformers) is an open-sourced NLP pre-training model developed by researchers at Google in 2018. A direct descendant to GPT (Generalized Language Models), BERT has outperformed several models in NLP and provided top results in Question Answering, Natural Language Inference (MNLI), and other frameworks.

What makes it's unique from the rest of the model is that it is the first deeply bidirectional, unsupervised language representation, pre-trained using only a plain text corpus. Since it's open-sourced, anyone with machine learning knowledge can easily build an NLP model without the need for sourcing massive datasets for training the model, thus saving time, energy, knowledge and resources.

How does it work?

Traditional context-free models (like word2vec or GloVe) generate a single word embedding representation for each word in the vocabulary which means the word “right” would have the same context-free representation in “I’m sure I’m right” and “Take a right turn.” However, BERT would represent based on both previous and next context, making it bidirectional. While the concept of bidirectional was around for a long time, BERT was first on its kind to successfully pre-train bidirectional in a deep neural network.

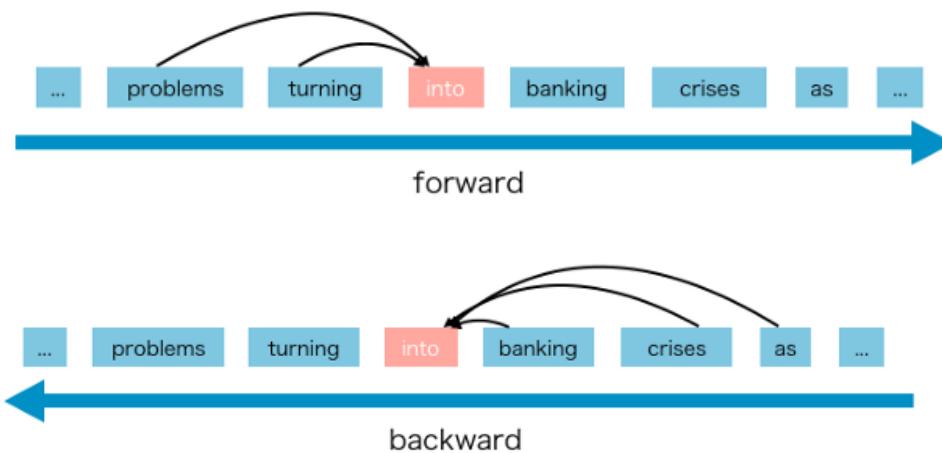


Q8.What is XLNet?

Answer:

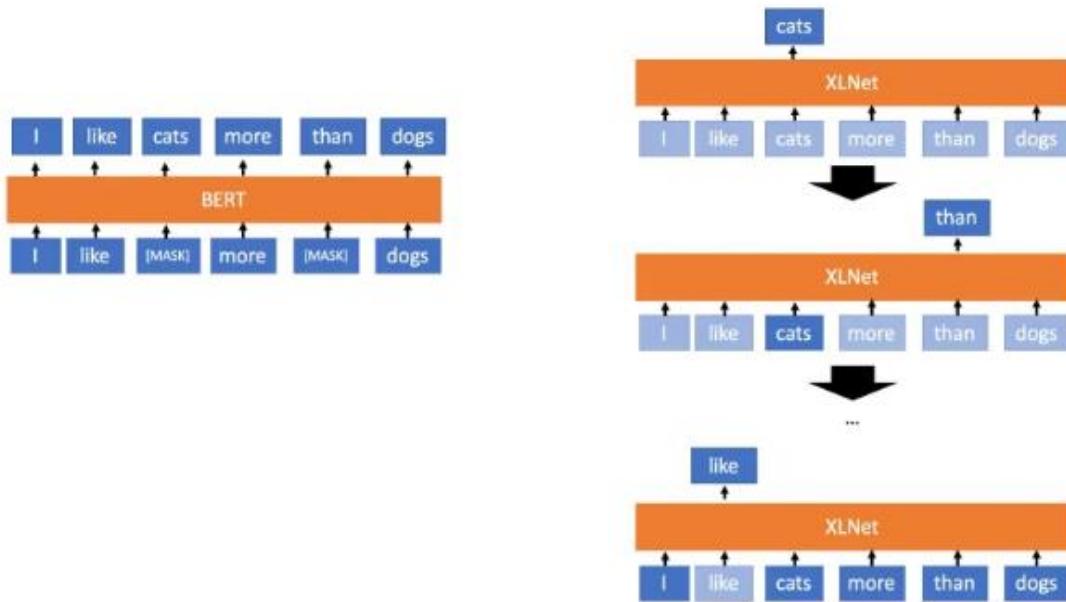
XLNet is a BERT-like model instead of a totally different one. But it is an auspicious and potential one. In one word, **XLNet is a generalized autoregressive pretraining method.**

Autoregressive (AR) language model: It is a kind of model that uses the context word to predict the next word. But here the context word is constrained to two directions, either forward or backwards.



The advantages of AR language model are good at generative Natural Language Process(NLP) tasks. Because when generating context, usually is the forward direction. AR language model naturally works well on such NLP tasks.

But Autoregressive language model has some disadvantages, and it only can use forward context or backward context, which means it can't use forward and backward context at the same time.



The conceptual difference between BERT and XLNet. Transparent words are masked out so the model cannot rely on them. XLNet learns to predict the words in an arbitrary order but in an autoregressive, sequential manner (not necessarily left-to-right). BERT predicts all masked words simultaneously.

Q9. What is the transformer?

[Answer:](#)

Transformer: It is a deep machine learning model introduced in 2017, used primarily in the field of natural language processing (NLP). Like recurrent neural networks(RNN), It is designed to handle ordered sequences of data, such as natural language, for various tasks like machine translation and text summarization. However, Unlike recurrent neural networks(RNN), Transformers do not require that the sequence be processed in the order. So, if the data in question is a natural language, the Transformer does not need to process the beginning of a sentence before it processes the end. Due to this feature, the Transformer allows for much more parallelization than RNNs during training.

Transformers are developed to solve the problem of sequence transduction current neural networks. It means any task that transforms an input sequence to an output sequence. This includes speech recognition, text-to-speech transformation, etc.

For models to perform a sequence transduction, it is necessary to have some sort of memory. example, let us say that we are translating the following sentence to another language (French):

“The Transformers” is a Japanese band. That band was formed in 1968, during the height of the Japanese music history.”

In the above example, the word “the band” in the second sentence refers to the band “The Transformers” introduced in the first sentence. When you read about the band in the second sentence, you know that it is referencing to the “The Transformers” band. That may be important for translation.

For translating other sentences like that, a model needs to figure out these sort of dependencies and connections. Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) have been used to deal with this problem because of their properties.

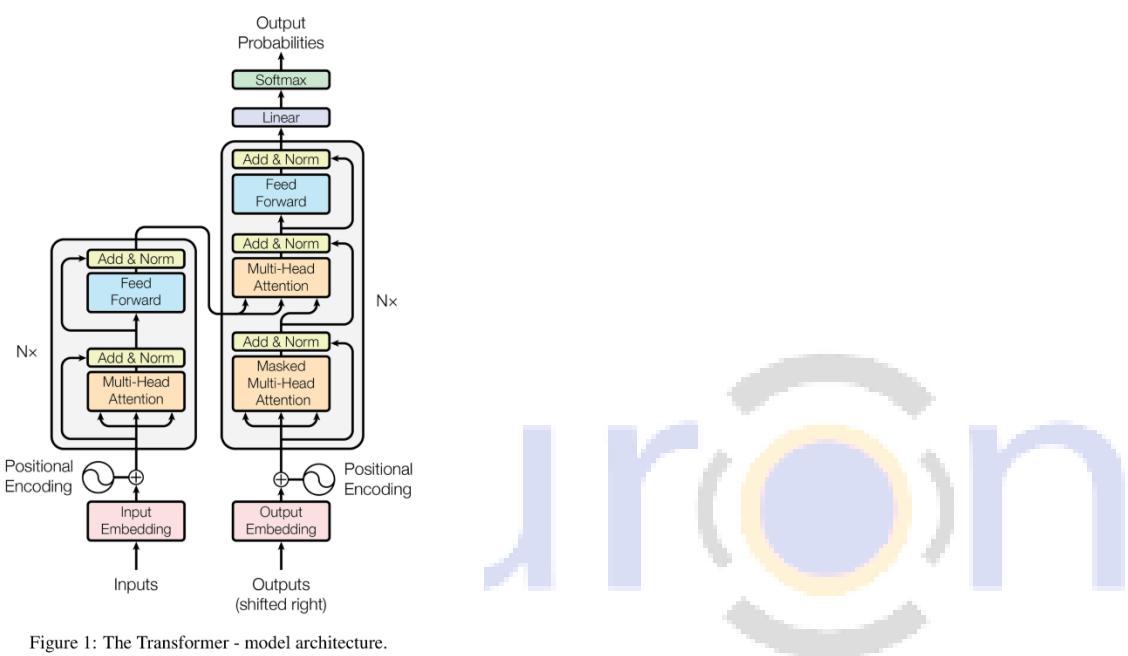


Figure 1: The Transformer - model architecture.

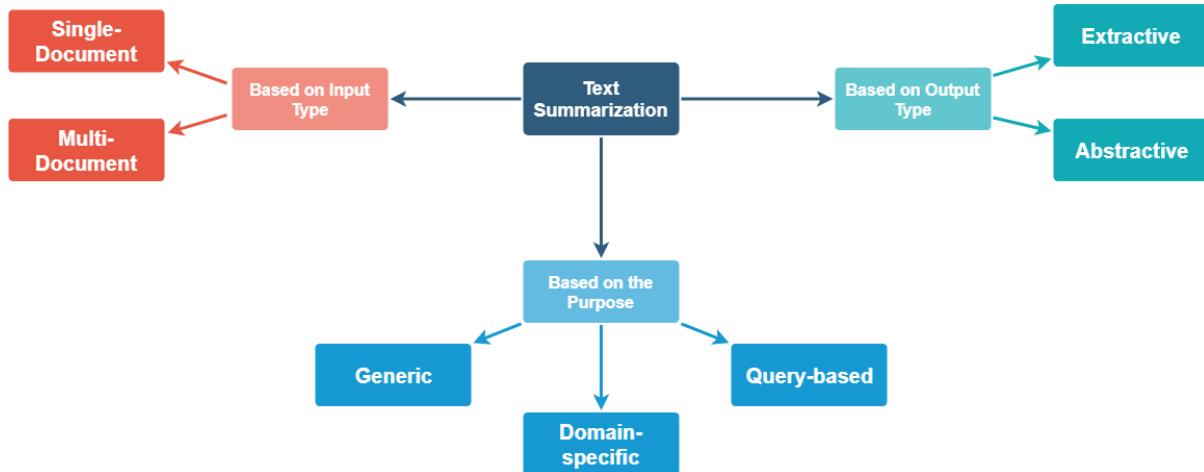
Q10. What is Text summarization?

Answer:

Text summarization: It is the process of shortening a text document, to create a summary of the significant points of the original document.

Types of Text Summarization Methods :

Text summarization methods can be classified into different types.

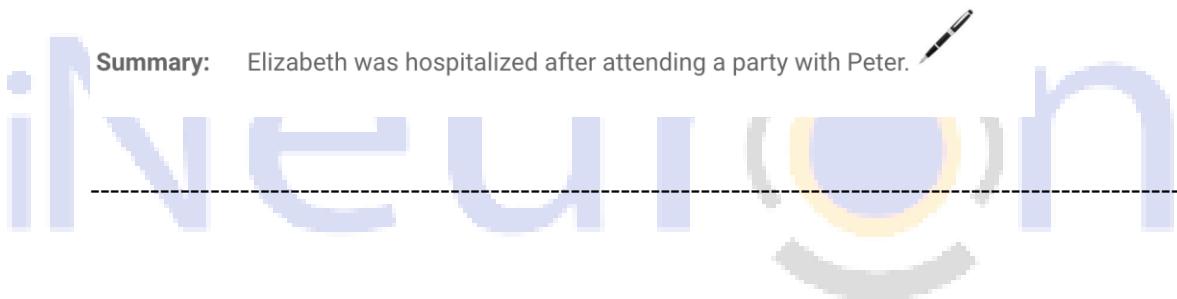


Example:

Source Text: Peter and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

Summary: Elizabeth was hospitalized after attending a party with Peter.



Q1. What is Levenshtein Algorithm?

Answer:

Levenshtein distance is a string metric for measuring the difference between two sequences. The Levenshtein distance between two words is the minimum number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other.

Mathematically, the Levenshtein distance between the two strings a , b (of length $|a|$ and $|b|$ respectively) is given by the formula, $\text{lev}_a, b(|a|, |b|)$ where :

$$\text{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \text{lev}_{a,b}(i-1, j) + 1 \\ \text{lev}_{a,b}(i, j-1) + 1 \\ \text{lev}_{a,b}(i-1, j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

Where, $1_{(a_i \neq b_j)}$: This is the indicator function equal to zero when $a_i \neq b_j$ and equal to 1 otherwise, and $\text{lev}_a, b(i,j)$ is the distance between the first i characters of a and the first j characters of b .

Example:

The Levenshtein distance between "HONDA" and "HYUNDAI" is 3, since the following three edits change one into the other, and there is no way to do it with fewer than three edits:

insertion
substitution
deletion

H		O	N	D	A	
H	Y	U	N	D	A	I

H	O		N	D	A	
H	Y	U	N	D	A	I

Q2. What is Soundex?

Answer:

Soundex attempts to find similar names or homophones using phonetic notation. The program retains letters according to detailed equations, to match individual titles for purposes of ample volume research.

Soundex phonetic algorithm: Its indexes strings depend on their English pronunciation. The algorithm is used to describe homophones, words that are pronounced the same, but spelt differently.

Suppose we have the following sourceDF.

```
+----+----+
|word1|word2|
+----+----+
|  to|  two|
|brake|break|
| here| hear|
| tree| free|
+----+----+
```

Let's run below code and see how the soundex algorithm encodes the above words.

```
val actualDF = sourceDF.withColumn(
  "w1_soundex",
  soundex(col("word1")))
  .withColumn(
  "w2_soundex",
  soundex(col("word2")))
)

actualDF.show()
```

```
+----+----+----+----+
|word1|word2|w1_soundex|w2_soundex|
+----+----+----+----+
|  to|  two|      T000|      T000|
|brake|break|      B620|      B620|
| here| hear|      H600|      H600|
| tree| free|      T600|      F600|
+----+----+----+----+
```

Let's summarize the above results:

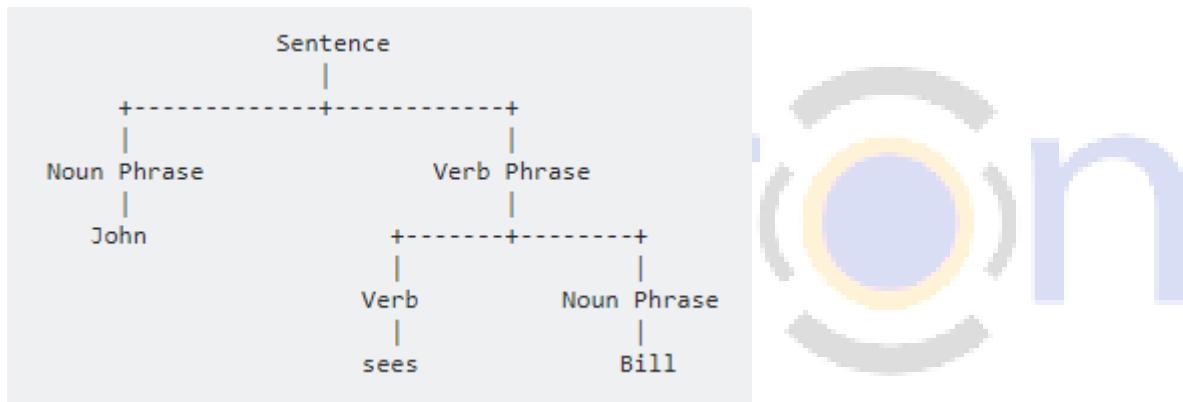
- "two" and "to" both are encoded as T000
- "break" and "brake" both are encoded as B620
- "hear" and "here" both are encoded as H600
- "free" is encoded as F600 and "tree" is encoded as T600: Encodings are similar, but word is different

The Soundex algorithm was often used to compare first names that were spelt differently.

Q3. What is Constituency parse?

[Answer:](#)

A constituency parse tree breaks a text into sub-phrases. Non-terminals in the tree are types of phrases, the terminals are the words in the sentence, and the edges are unlabeled. For a simple sentence, "John sees Bill", a constituency parse would be:



Above approaches convert the parse tree into a sequence following a depth-first traversal to be able to apply sequence-to-sequence models to it. The linearized version of the above parse tree looks as follows: (S (N) (VP V N)).

Q4. What is LDA(Latent Dirichlet Allocation)?

[Answer:](#)

LDA: It is used to classify text in the document to a specific topic. LDA builds a topic per document model and words per topic model, modelled as Dirichlet distributions.

- Each document is modeled as a distribution of topics, and each topic is modelled as multinomial distribution of words.
- LDA assumes that every chunk of text we feed into it will contain words that are somehow related. Therefore choosing the right corpus of data is crucial.

- It also assumes documents are produced from a mixture of topics. Those topics then generate words based on their probability distribution.

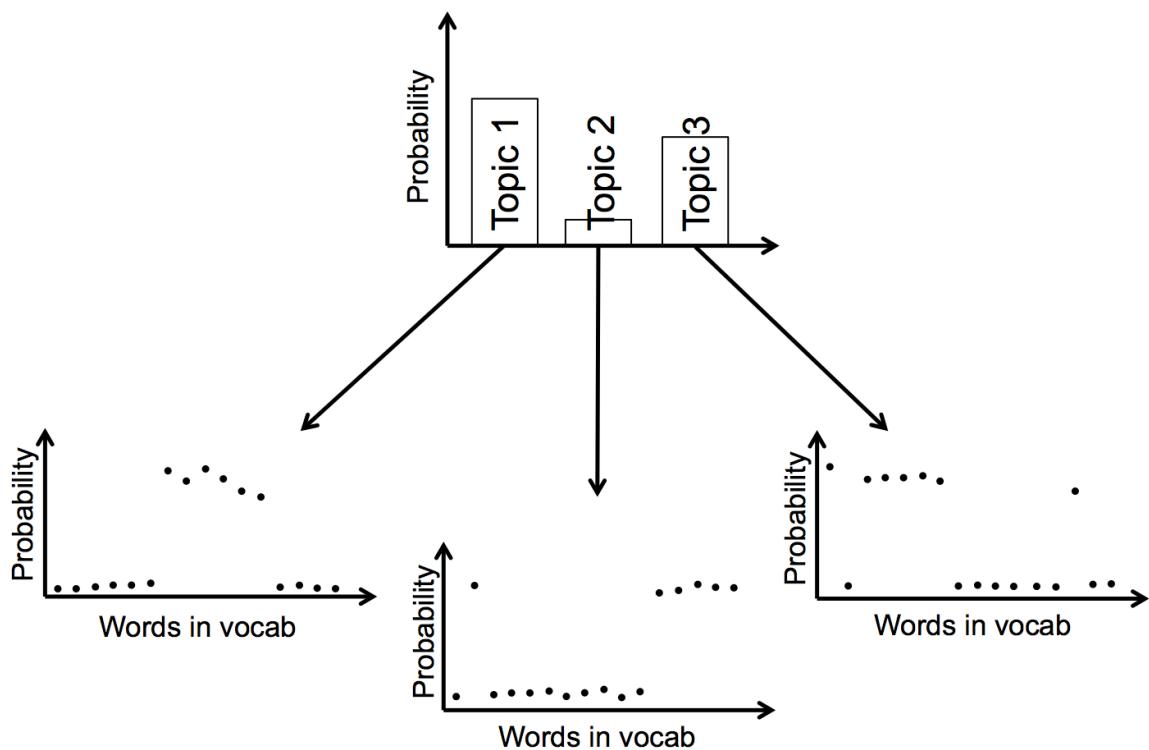
The Bayesian version of PLSA is LDA. It uses Dirichlet priors for the word-topic and document-topic distributions, lending itself to better generalization.

What LDA give us?

It is a probabilistic method. For every document, the results give us a mixture of topics that make up the document. To be precise, we can get probability distribution over the k topics for every document. Every word in the document is attributed to the particular topic with probability given by distribution.

These topics themselves were defined as probability distributions over vocabulary. Our results are two sets of probability distributions:

- The collection of distributions of topics for each document
- The collection of distributions of words for each topic.



Q5.What is LSA?

Answer:

Latent Semantic Analysis (LSA): It is a theory and the method for extract and represents the contextual usage meaning of words by statistical computation applied to large corpus of texts.

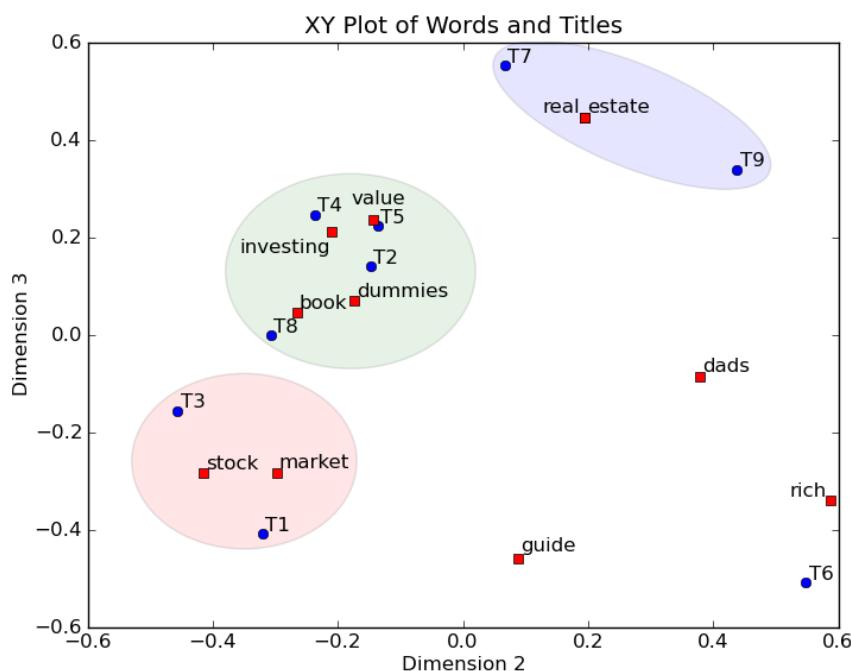
It is an information retrieval technique which analyzes and identifies the pattern in an unstructured collection of text and relationship between them.

Latent Semantic Analysis itself is an unsupervised way of uncovering synonyms in a collection of documents.

Why LSA(Latent Semantic Analysis)?

LSA is a technique for creating vector representation of the document. Having a vector representation of the document gives us a way to compare documents for their similarity by calculating the distance between vectors. In turn, means we can do handy things such as classify documents to find out which of a set knows topics they most likely reside to.

Classification implies we have some known topics that we want to group documents into, and that you have some labelled training data. If you're going to identify natural groupings of the documents without any labelled data, you can use clustering



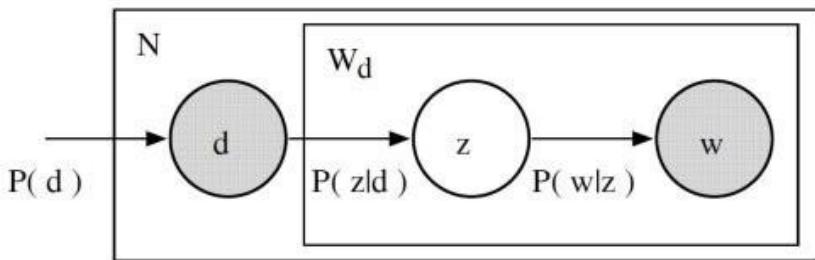
Q6. What is PLSA?

Answer:

PLSA stands for Probabilistic Latent Semantic Analysis, uses a probabilistic method instead of SVD to tackle problem. The main idea is to find the probabilistic model with latent topics that we can *generate* data we observe in our document term matrix. Specifically, we want a model $P(D, W)$ such that for any document d and word w , $P(d, w)$ corresponds to that entry in document-term matrix.

Each document is found in the mixture of topics, and each topic consists of the collection of words. PLSA adds the probabilistic spin to these assumptions:

- Given document d , topic z is available in that document with the probability $P(z|d)$
- Given the topic z , word w is drawn from z with probability $P(w|z)$



The joint probability of seeing the given document and word together is:

$$P(D, W) = P(D) \sum_Z P(Z|D)P(W|Z)$$

In the above case, $P(D)$, $P(Z|D)$, and $P(W|Z)$ are the parameters of our models. $P(D)$ can be determined directly from corpus. $P(Z|D)$ and the $P(W|Z)$ are modelled as multinomial distributions and can be trained using the expectation-maximisation algorithm (EM).

Q7. What is LDA2Vec?

Answer:

It is inspired by LDA, word2vec model is expanded to simultaneously learn word, document, topic and paragraph topic vectors.

Lda2vec is obtained by modifying the skip-gram word2vec variant. In the original skip-gram method, the model is trained to predict context words based on a pivot word. In lda2vec, the pivot word vector and a document vector are added to obtain a context vector. This context vector is then used to predict context words.

At the document level, we know how to represent the text as mixtures of topics. At the word-level, we typically used something like word2vec to obtain vector representations. It is an extension of word2vec and LDA that jointly learns word, document, and topic vectors.

How does it work?

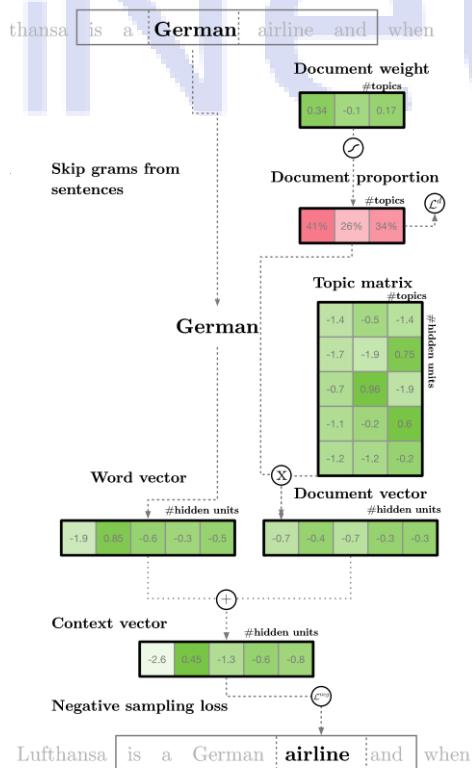
It correctly builds on top of the skip-gram model of word2vec to generate word vectors. Neural net that learns word embedding by trying to use input word to predict enclosing context words.

With Lda2vec, other than using the word vector directly to predict context words, you leverage a context vector to make the predictions. Context vector is created as the sum of two other vectors: the word vector and the document vector.

The same skip-gram word2vec model generates the word vector. The document vector is most impressive. It is a really weighted combination of two other components:

- the document weight vector, representing the “weights” of each topic in a document
- Topic matrix represents each topic and its corresponding vector embedding.

Together, a document vector and word vector generate “context” vectors for each word in a document. Lda2vec power lies in the fact that it not only learns word embeddings for words; it simultaneously learns topic representations and document representations as well.



Q8. What is Expectation-Maximization Algorithm(EM)?

Answer:

The Expectation-Maximization Algorithm, in short, EM algorithm, is an approach for maximum likelihood estimation in the presence of latent variables.

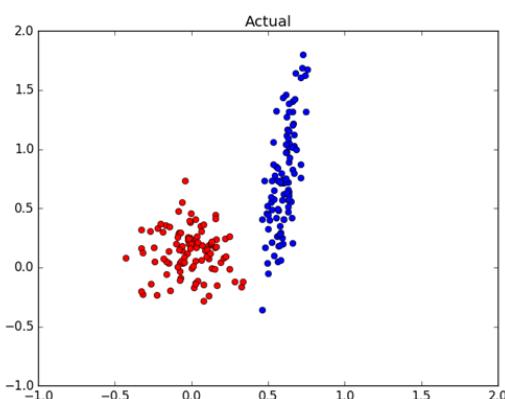
This algorithm is an iterative approach that cycles between two modes. The first mode attempts to predict the missing or latent variables called the estimation-step or E-step. The second mode attempts to optimise the parameters of the model to explain the data best called the maximization-step or M-step.

- **E-Step.** Estimate the missing variables in the dataset.
- **M-Step.** Maximize the parameters of the model in the presence of the data.

The EM algorithm can be applied quite widely, although it is perhaps most well known in machine learning for use in unsupervised learning problems, such as density estimation and clustering.

For detail explanation of EM is, let us first consider this example. Say that we are in a school, and interested to learn the height distribution of female and male students in the school. The most sensible thing to do, as we probably would agree with me, is to randomly take a sample of N students of both genders, collect their height information and estimate the mean and standard deviation for male and female separately by way of maximum likelihood method.

Now say that you are not able to know the gender of student while we collect their height information, and so there are two things you have to guess/estimate: (1) whether the individual sample of height information belongs to a male or a female and (2) the parameters (μ, θ) for each gender which is now unobservable. This is tricky because only with the knowledge of who belongs to which group, can we make reasonable estimates of the group parameters separately. Similarly, only if we know the parameters that define the groups, can we assign a subject properly. How do you break out of this infinite loop? Well, EM algorithm just says to start with initial random guesses.



Q9.What is Text classification in NLP?

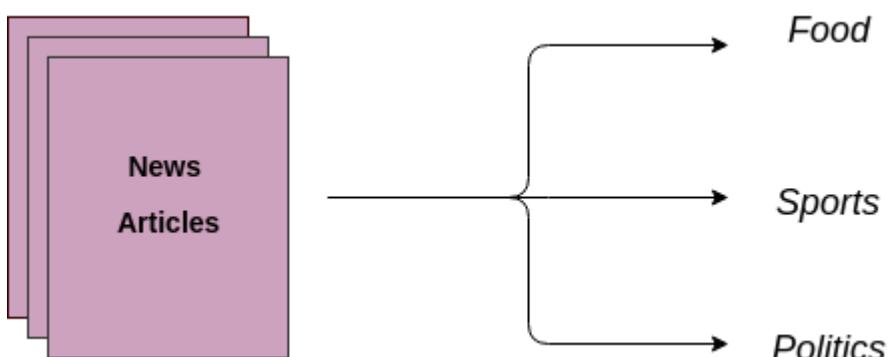
Answer:

Text classification is also known as text tagging or text categorization is a process of categorizing text into organized groups. By using NLP, text classification can automatically analyze text and then assign a set of pre-defined tags or categories based on content.

Unstructured text is everywhere on the internet, such as emails, chat conversations, websites, and the social media but it's hard to extract value from given data unless it's organized in a certain way. Doing so used to be a difficult and expensive process since it required spending time and resources to manually sort the data or creating handcrafted rules that are difficult to maintain. Text classifiers with NLP have proven to be a great alternative to structure textual data in a fast, cost-effective, and scalable way.

Text classification is becoming an increasingly important part of businesses as it allows us to get insights from data and automate business processes quickly. Some of the most common examples and the use cases for automatic text classification include the following:

- **Sentiment Analysis:** It is the process of understanding if a given text is talking positively or negatively about a given subject (e.g. for brand monitoring purposes).
- **Topic Detection:** In this, the task of identifying the theme or topic of a piece of text (e.g. know if a product review is about Ease of Use, Customer Support, or Pricing when analysing customer feedback).
- **Language Detection:** the procedure of detecting the language of a given text (e.g. know if an incoming support ticket is written in English or Spanish for automatically routing tickets to the appropriate team).



Q10. What is Word Sense Disambiguation (WSD)?

Answer:

WSD (Word Sense Disambiguation) is a solution to the ambiguity which arises due to different meaning of words in a different context.

In natural language processing, **word sense disambiguation** (WSD) is the problem of determining which "sense" (meaning) of a word is activated by the use of the word in a particular context, a process which appears to be mostly unconscious in people. WSD is the natural classification problem: Given a word and its possible senses, as defined by the dictionary, classify an occurrence of the word in the context into one or more of its sense classes. The features of the context (such as the neighbouring words) provide the evidence for classification.

For example, consider these two below sentences.

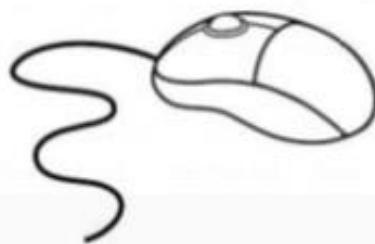
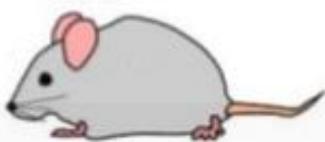
“ The **bank** will not be accepting the cash on Saturdays. ”

“ The river overflowed the **bank** .”

The word “ **bank** ” in the given sentence refers to commercial (finance) banks, while in the second sentence, it refers to a riverbank. The uncertainty that arises, due to this is tough for the machine to detect and resolve. Detection of change is the first issue and fixing it and displaying the correct output is the second issue.

Word Sense disambiguation

I need new batteries for my **mouse**.



Q1. What is LSI(Latent Semantic Indexing)?

Answer:

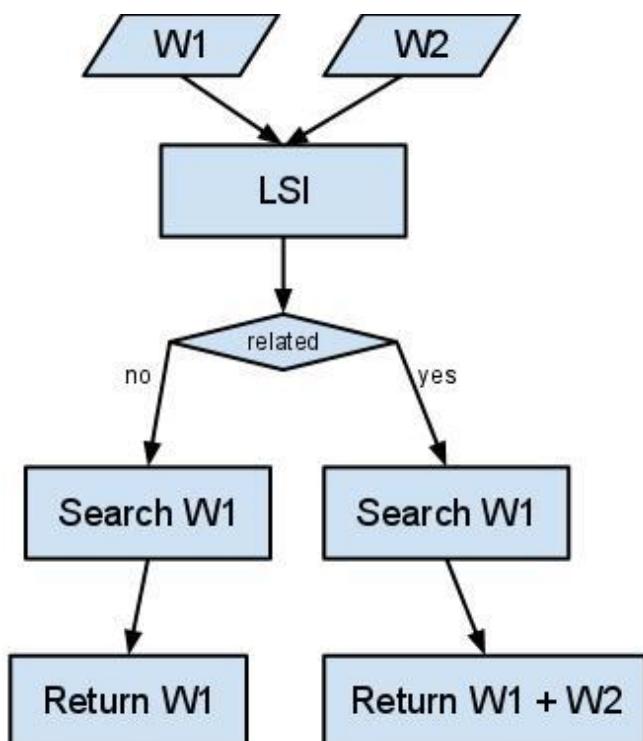
Latent Semantic Indexing (LSI): It is an indexing and retrieval method that uses a mathematical technique called SVD(Singular value decomposition) to find patterns in relationships between terms and concepts contained in an unstructured collection of text. It is based on the principle that words that are used in the same contexts tend to have similar meanings.

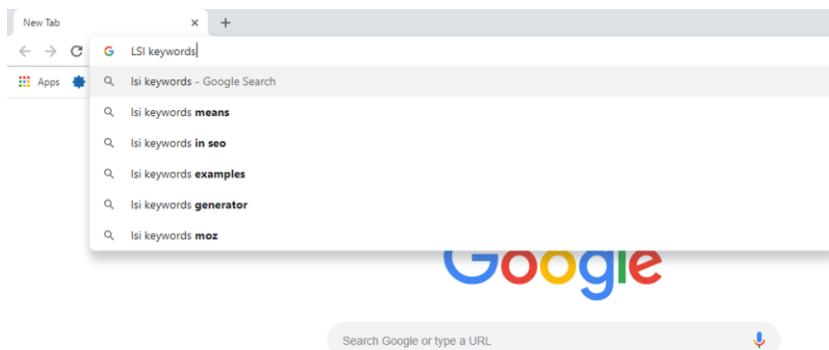
For example, Tiger and Woods are associated with men instead of an animal, and a Wood, Parris, and Hilton are associated with the singer.

Example:

If you use LSI to index a collection of articles and the words “fan” and “regulator” appear together frequently enough, the search algorithm would notice that the two terms are semantically close. A search for “fan” will, therefore, return a set of items containing that phrase, but also items that contain just the word “regulator”. It doesn't understand word distance, but by examining a sufficient number of documents, it only knows the two terms are interrelated. It then uses that information to provide an expanded set of results with better recall than an understandable keyword search.

The diagram below describes the effect between LSI and keyword searches. W stands for a document.





Q2. What is Named Entity Recognition? And tell some use cases of NER?

Answer:

Named-entity recognition (NER): It is also known as entity extraction, and entity identification is a subtask of information extraction that explores to locate and classify atomic elements in text into predefined categories like the names of persons, organizations, places, expressions of times, quantities, monetary values, percentages and more.

In each text document, particular terms represent specific entities that are more informative and have a different context. These entities are called named entities, which more accurately refer to conditions that represent real-world objects like people, places, organizations or institutions, and so on, which are often expressed by proper names. The naive approach could be to find these by having a look at the noun phrases in text documents. It also is known as entity chunking/extraction, which is a popular technique used in information extraction to analyze and segment the named entities and categorize or classify them under various predefined classes.

Named Entity Recognition use-case

- **Classifying content for news providers-**

NER can automatically scan entire articles and reveal which are the significant people, organizations, and places discussed in them. Knowing the relevant tags for each item helps in automatically categorizing the articles in defined hierarchies and enable smooth content discovery.

- **Customer Support:**

Let's say we are handling the customer support department of an electronics store with multiple branches worldwide; we go through a number of mentions in our customers' feedback. Such as this for instance.

Now, if we pass it through the Named Entity Recognition API, it pulls out the entities Bangalore (location) and Fitbit (Product). This can be then used to categorize the complaint and assign it to the relevant department within the organization that should be handling this.



Figure 1: An example of NER application on an example text

Q3. What is perplexity?

Answer:

Perplexity: It is a measurement of how well a probability model predicts a sample. In the context of NLP, perplexity(Confusion) is one way to evaluate language models.

The term perplexity has three closely related meanings. It is a measure of how easy a probability distribution is to predict. It is a measure of how variable a prediction model is. And It is a measure of prediction error. The third meaning of perplexity is calculated slightly differently, but all three have the same fundamental idea.

Dan Jurafsky



Perplexity

The best language model is one that best predicts an unseen test set

- Gives the highest $P(\text{sentence})$

Perplexity is the inverse probability of the test set, normalized by the number of words:

$$PP(W) = P(w_1 w_2 \dots w_N)^{\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

Chain rule:

$$PP(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_1 \dots w_{i-1})}}$$

For bigrams:

$$PP(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_{i-1})}}$$

Minimizing perplexity is the same as maximizing probability

Q4. What is the language model?

Answer:

Language Modelling (LM): It is one of the essential parts of modern NLP. There are many sorts of applications for Language Modelling, like Machine Translation, Spell Correction Speech Recognition, Summarization, Question Answering, Sentiment analysis, etc. Each of those tasks requires the use of the language model. The language model is needed to represent the text to a form understandable from the machine point of view.

The statistical language model is a probability distribution over a series of words. Given such a series, say of length m, it assigns a probability to the whole series.

It provides context to distinguish between phrases and words that sound similar. For example, in American English, the phrases "wreck a nice beach" and "recognize speech" sound alike but mean different things.

Data sparsity is a significant problem in building language models. Most possible word sequences are not noticed in training. One solution is to make the inference that the probability of a word only depends on the previous n words. This is called as an n -gram model or unigram model when $n = 1$. The unigram model is also known as the bag of words model.

How does this Language Model help in NLP Tasks?

The probabilities restoration by a language model is most useful to compare the likelihood that different sentences are "good sentences." This was useful in many practical tasks, for example:

Spell checking: You observe a word that is not identified as a known word as part of a sentence. Using the edit distance algorithm, we find the closest known words to the unknown words. These are the candidate corrections. For example, we observe the word "wurd" in the context of the sentence, "I like to write this wurd." The candidate corrections are ["word", "weird", "wind"]. How can we select among these candidates the most likely correction for the suspected error "weird"?

Automatic Speech Recognition: we receive as input a string of phonemes; a first model predicts for sub-sequences of the stream of phonemes candidate words; the language model helps in ranking the most likely sequence of words compatible with the candidate words produced by the acoustic model.

Machine Translation: each word from the source language is mapped to multiple candidate words in the target language; the language model in the target language can rank the most likely sequence of candidate target words.

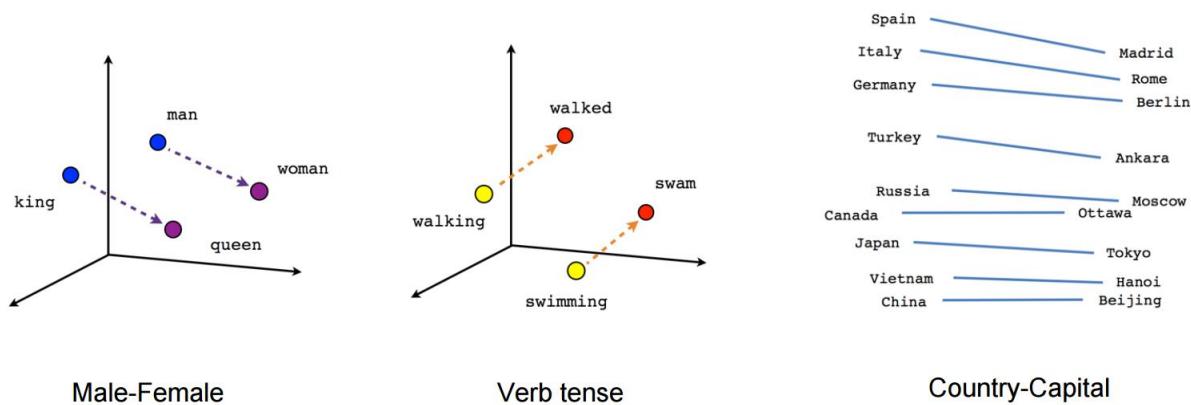
Q5. What is Word Embedding?

Answer:

A word embedding is a learned representation for text where words that have the same meaning have a similar observation.

It is basically a form of word representation that bridges the human understanding of language to that of a machine. Word embeddings divide representations of text in an n-dimensional space. These are essential for solving most NLP problems.

And the other point worth considering is how we obtain word embeddings as no two sets of word embeddings are similar. Word embeddings aren't random; they're developed by training the neural network. A recent powerful word embedding usage comes from Google named Word2Vec, which is trained by predicting several words that appear next to other words in a language. For example, the word "cat", the neural network would predict the words like "kitten" and "feline." This intuition of words comes out "near" each other allows us to place them in vector space.



Q6. Do you have an idea about fastText?

Answer:

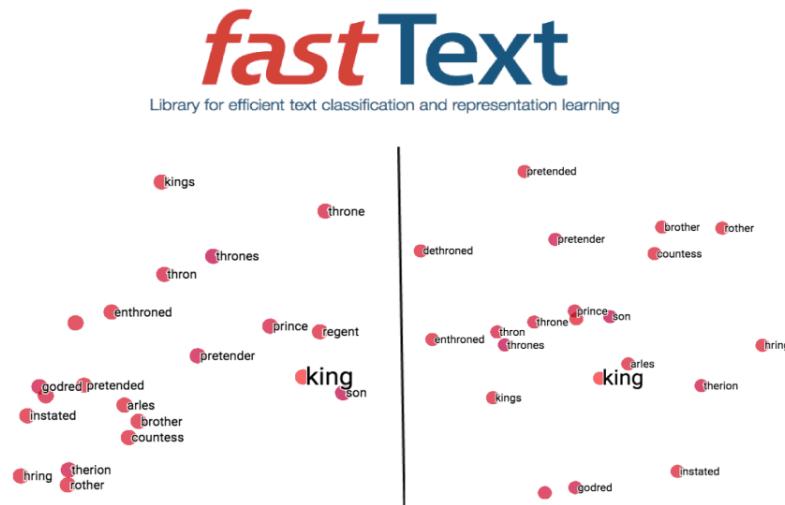
fastText: It is another word embedding method that is an extension of the word2vec model. Alternatively, learning vectors for words directly. It represents each word as an n-gram of characters. So, for example, take the word, “artificial” with n=3, the fastText representation of this word is <ar, art, rti, tif, ifi, fic, ici, ial, al>, where the angular brackets indicate the beginning and end of the word.

This helps to capture the meaning of shorter words and grant the embeddings to understand prefixes and suffixes. Once the word has been showed using character skip-grams, a n-gram model is trained to learn the embeddings. This model is acknowledged to be a bag of words model with a sliding

window over a word because no internal structure of the word is taken into account. As long as the characters are within this window, the order of the n-grams doesn't matter.

fastText works well with rare words. So even if a word wasn't seen during training, it can be broken down into n-grams to get its embeddings.

Word2vec and GloVe both fail to provide any vector representation for words that are not in the model dictionary. This is a huge advantage of this method.



Q7. What is GloVe?

Answer:

GloVe(global vectors) is for word representation. GloVe is an unsupervised learning algorithm developed by Stanford for achieving word embeddings by aggregating a global word-word co-occurrence matrix from a corpus. The resulting embeddings show interesting linear substructures of the word in vector space.

The GloVe model produces a vector space with meaningful substructure, as evidenced by its performance of 75% on a new word analogy task. It also outperforms related models on similarity tasks and named entity recognition.

How GloVe find meaning in statistics?

Produces a vector space with meaningful substructure, as evidenced by its performance of 75% on a new word analogy task. It also outperforms related models on similarity tasks and named entity recognition.

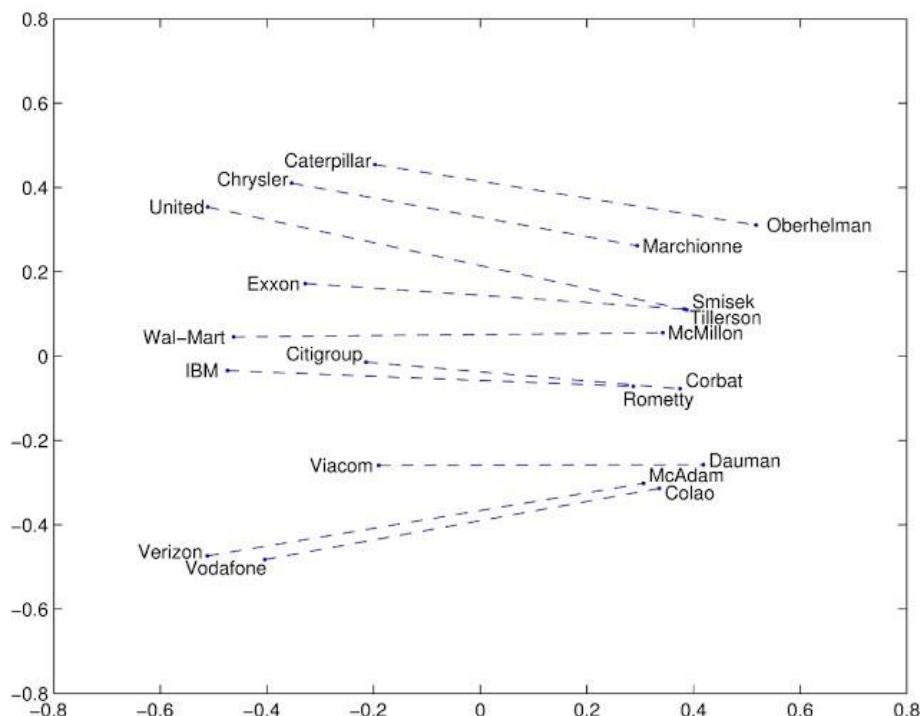
GloVe aims to achieve two goals:

- (1) Create word vectors that **capture meaning in vector space**
- (2) Takes advantage of **global count statistics** instead of only local information

Unlike word2vec – which learns by streaming sentences – GloVe determines based on a **co-occurrence matrix** and trains word vectors, so their differences predict **co-occurrence ratios**

GloVe weights the loss based on word frequency.

Somewhat surprisingly, word2vec and GloVe turn out to be remarkably similar, despite starting off from entirely different starting points.



Q8. Explain Gensim?

Answer:

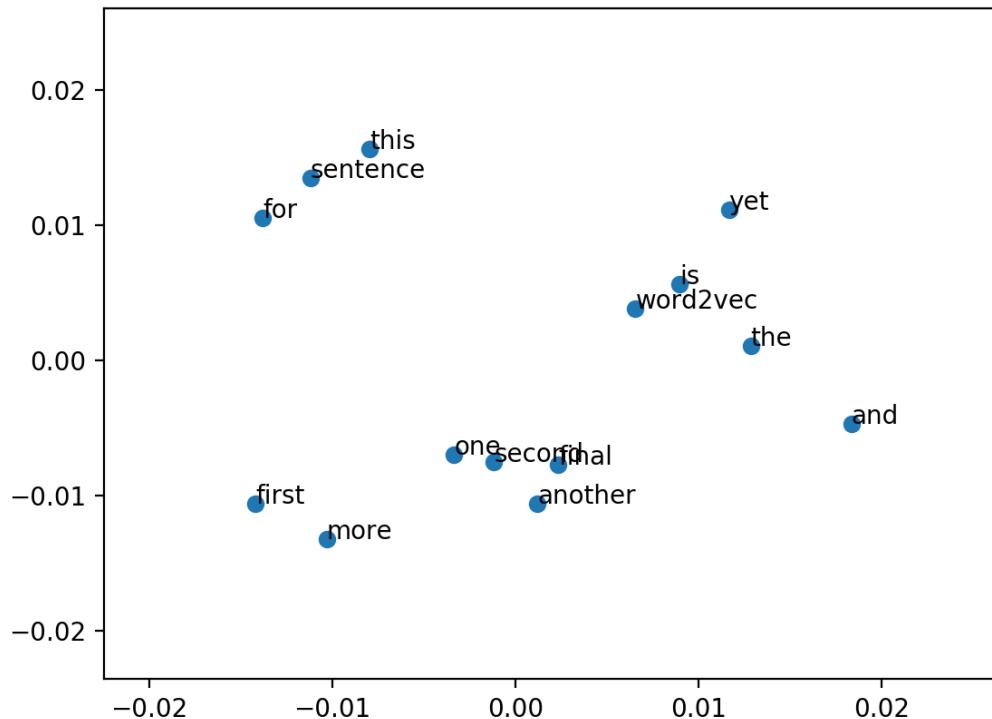
Gensim: It is billed as a Natural Language Processing package that does ‘Topic Modeling for Humans’. But its practically much more than that.

If you are unfamiliar with topic modeling, it is a technique to extract the underlying topics from large volumes of text. Gensim provides algorithms like LDA and LSI (which we already seen in previous interview questions) and the necessary sophistication to built high-quality topic models.

It is an excellent library package for processing texts, working with word vector models (such as FastText, Word2Vec, etc) and for building the topic models. Another significant advantage with gensim is: it lets us handle large text files without having to load the entire file in memory.

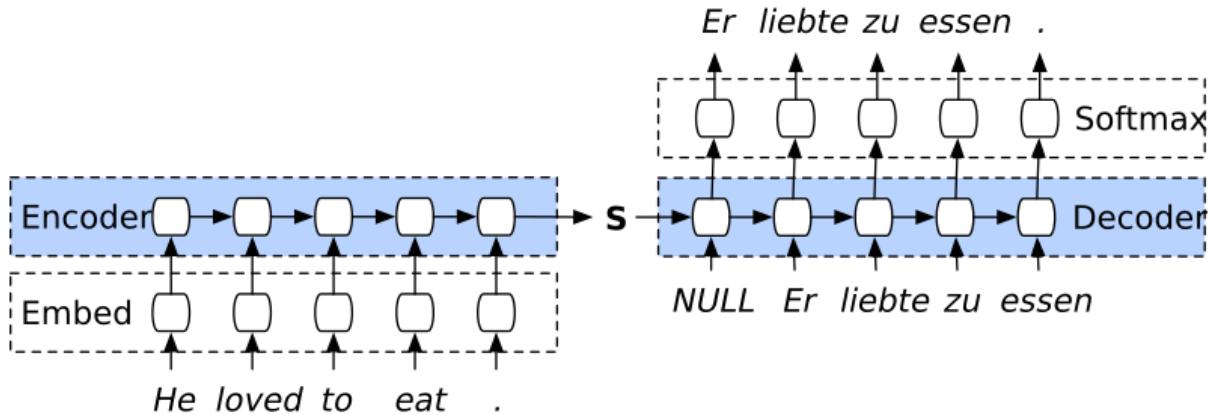
We can also tell as It is an open-source library for unsupervised topic modeling and natural language processing, using modern statistical machine learning.

Gensim is implemented in Python and Cython. Gensim is designed to handle extensive text collections using data streaming and incremental online algorithms, which differentiates it from most other machine learning software packages that target only in-memory processing.



Q9. What is Encoder-Decoder Architecture?

Answer:



The encoder-decoder architecture consists of two main parts :

- **Encoder:**

Encoder simply takes the input data, and trains on it, then it passes the final state of its recurrent layer as an initial state to the first recurrent layer of the decoder part.

```
Encoder input : English sentences
```

```
Encoder initial state : It depends on the initializer we use
```

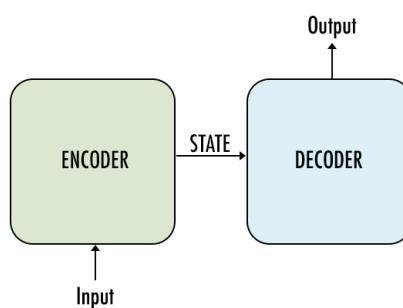
- **Decoder :**

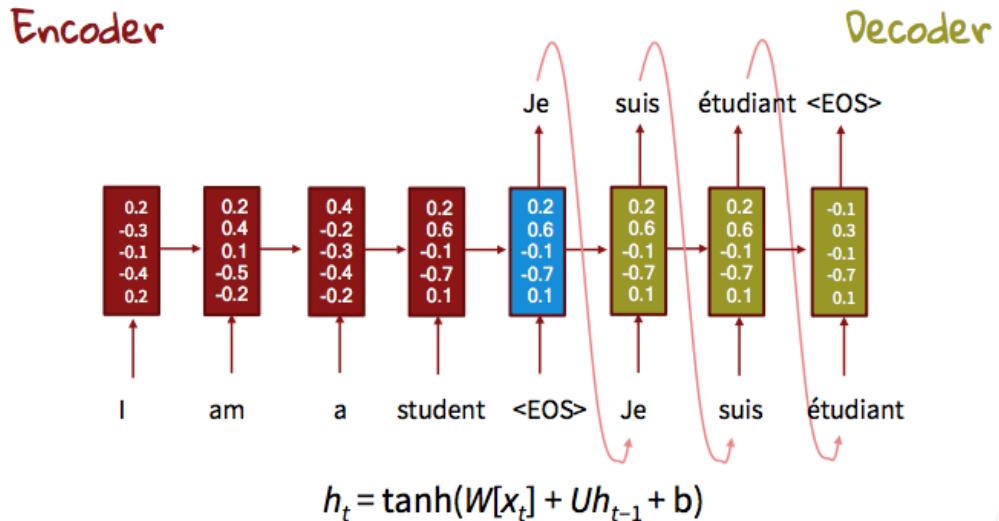
The decoder takes the final state of encoder's final recurrent layer and uses it as an initial state to its initial, recurrent layer, the input of the decoder is sequences that we want to get French sentences.

```
Decoder input : French sentences
```

```
Decoder initial state : The last state of encoder's last recurrent layer
```

Some more example for better understanding:



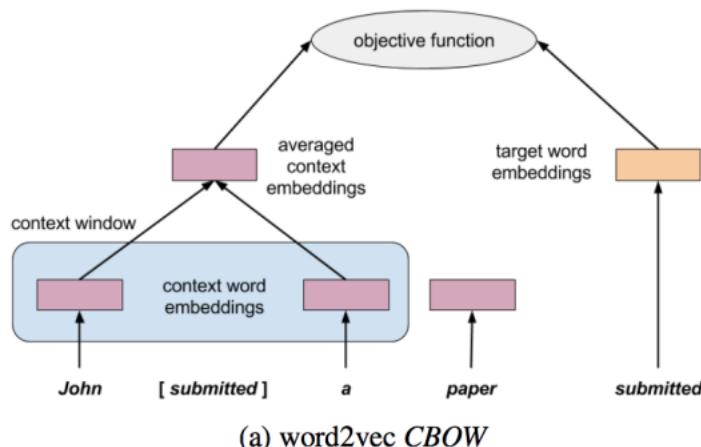


Q10. What is Context2Vec?

Answer:

Assume a case where you have a sentence like. I can't find May. Word May maybe refers to a month's name or a person's name. You use the words surround it (context) to help yourself to determine the best suitable option. Actually, this problem refers to the Word Sense Disambiguation task, on which you investigate the actual semantics of the word based on several semantic and linguistic techniques. The Context2Vec idea is taken from the original CBOW Word2Vec model, but instead of relying on averaging the embedding of the words, it relies on a much more complex parametric model that is based on one layer of Bi-LSTM. Figure1 shows the architecture of the CBOW model.

Figure1



Context2Vec applied the same concept of windowing, but instead of using a simple average function, it uses 3 stages to learn complex parametric networks.

- A Bi-LSTM layer that takes left-to-right and right-to-left representations
- A feedforward network that takes the concatenated hidden representation and produces a hidden representation through learning the network parameters.
- Finally, we apply the objective function to the network output.

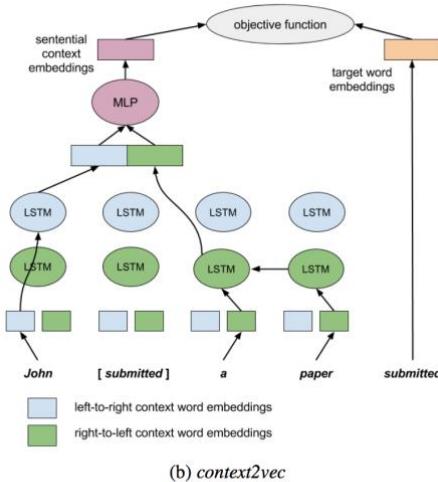


Figure 1: *word2vec* and *context2vec* architectures.

We used the *Word2Vec* negative sampling idea to get better performance while calculating the loss value.

The following are some samples of the closest words to a given context.

Sentential Context	Closest target words
This [] is due	item, fact-sheet, offer, pack, card
This [] is due not just to mere luck	offer, suggestion, announcement, item, prize
This [] is due not just to mere luck, but to outstanding work and dedication	award, prize, turnabout, offer, gift
[] is due not just to mere luck, but to outstanding work and dedication	it, success, this, victory, prize-money

Table 1: Closest target words to various sentential contexts, illustrating *context2vec*'s sensitivity to long range dependencies, and both sides of the target word.

Q1. Do you have any idea about Event2Mind in NLP?

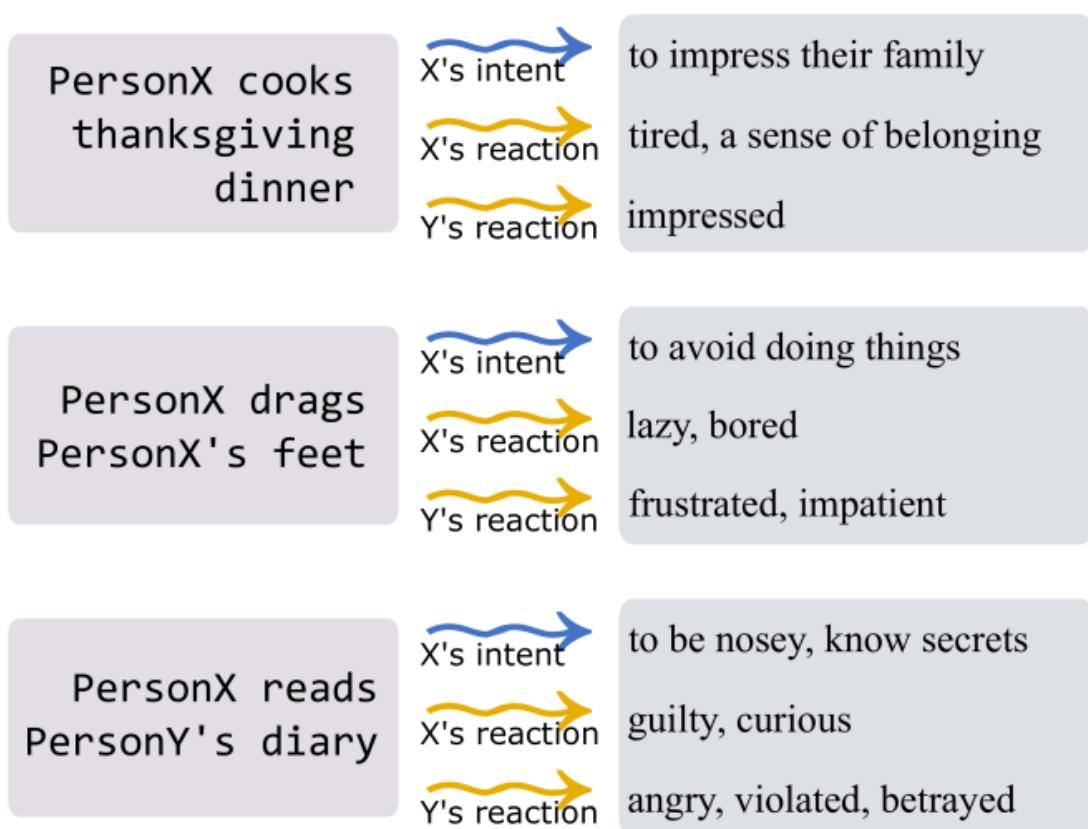
Answer:

Yes, it is based on NLP research paper to understand the common-sense inference from sentences.

Event2Mind: Common-sense Inference on Events, Intents, and Reactions

The study of “Commonsense Reasoning” in NLP deals with teaching computers how to gain and employ common sense knowledge. NLP systems require common sense to adapt quickly and understand humans as we talk to each other in a natural environment.

This paper proposes a new task to teach systems commonsense reasoning: given an event described in a short “event phrase” (e.g. “PersonX drinks coffee in the morning”), the researchers teach a system to reason about the likely intents (“PersonX wants to stay awake”) and reactions (“PersonX feels alert”) of the event’s participants.



Understanding a narrative requires common-sense reasoning about the mental states of people in relation to events. For example, if “Robert is dragging his feet at work,” pragmatic implications about Robert’s *intent* are that “Robert wants to avoid doing things” (Above Fig). You can also infer that Robert’s *emotional reaction* might be feeling “bored” or “lazy.” Furthermore, while not explicitly mentioned, you can assume that people other than Robert are affected by the situation, and these people are likely to feel “impatient” or “frustrated.”

This type of pragmatic inference can likely be useful for a wide range of NLP applications that require accurate anticipation of people's intents and emotional reactions, even when they are not expressly mentioned. For example, an ideal dialogue system should react in empathetic ways by reasoning about the human user's mental state based on the events the user has experienced, without the user explicitly stating how they are feeling. Furthermore, advertisement systems on social media should be able to reason about the emotional reactions of people after events such as mass shootings and remove ads for guns, which might increase social distress. Also, the pragmatic inference is a necessary step toward automatic narrative understanding and generation. However, this type of commonsense social reasoning goes far beyond the widely studied entailment tasks and thus falls outside the scope of existing benchmarks.

Q2. What is SWAG in NLP?

Answer:

SWAG stands for **Situations with Adversarial Generations** is a dataset consisting of 113k multiple-choice questions about a rich spectrum of grounded situations.

Swag: A Large Scale Adversarial Dataset for Grounded Commonsense Inference

According to NLP research paper on SWAG is “Given a partial description like “he opened the hood of the car,” humans can reason about the situation and anticipate what might come next (“then, he examined the engine”). In this paper, you introduce the task of grounded commonsense inference, unifying natural language inference(NLI), and common-sense reasoning.

We present SWAG, a dataset with 113k multiple-choice questions about the rich spectrum of grounded positions. To address recurring challenges of annotation artifacts and human biases found in many existing datasets, we propose AF(Adversarial Filtering), a novel procedure that constructs a de-biased dataset by iteratively training an ensemble of stylistic classifiers, and using them to filter the data. To account for the aggressive adversarial filtering, we use state-of-the-art language models to oversample a diverse set of potential counterfactuals massively. Empirical results present that while humans can solve the resulting inference problems with high accuracy (88%), various competitive models make an effort on our task. We provide a comprehensive analysis that indicates significant opportunities for future research.

When we read a tale, we bring to it a large body of implied knowledge about the physical world. For instance, given the context “on stage, a man takes a seat at the piano,” we can easily infer what the situation might look like: a man is giving a piano performance, with a crowd watching him. We can furthermore infer his likely next action: he will most likely set his fingers on the piano key and start playing.

This type of natural language inference(NLI) requires common-sense reasoning, substantially broadening the scope of prior work that focused primarily on linguistic entailment. Whereas the

dominant entailment paradigm asks if 2 natural language sentences (the ‘premise’ and the ‘hypothesis’) describe the same set of possible worlds, here we focus on whether a (multiple-choice) ending represents a possible (*future*) world that can arise from the situation described in the premise, even when it is not strictly entailed. Making such inference necessitates a rich understanding of everyday physical conditions, including object affordances and frame semantics.

On stage, a woman takes a seat  at the piano. She

- a) sits on a bench as her sister plays with the doll.
- b) smiles with someone as the music plays.
- c) is in the crowd, watching the dancers.
- d) nervously sets her fingers on the keys.**



A girl is going across a set of monkey bars. She

- a) jumps up across the monkey bars.
- b) struggles onto the monkey bars to grab her head.
- c) gets to the end and stands on a wooden plank.**
- d) jumps up and does a back flip.

The woman is now blow drying the dog. The dog

- a) is placed in the kennel next to a woman's feet.**
- b) washes her face with the shampoo.
- c) walks into frame and walks towards the dog.
- d) tried to cut her face, so she is trying to do something very close to her face.

Table 1: Examples from Swag; the correct answer is **bolded**. Adversarial Filtering ensures that stylistic models find all options equally appealing.

Q3. What is the Pix2Pix network?

Answer:

Pix2Pix network: It is a Conditional GANs (cGAN) that learn the mapping from an input image to output an image.

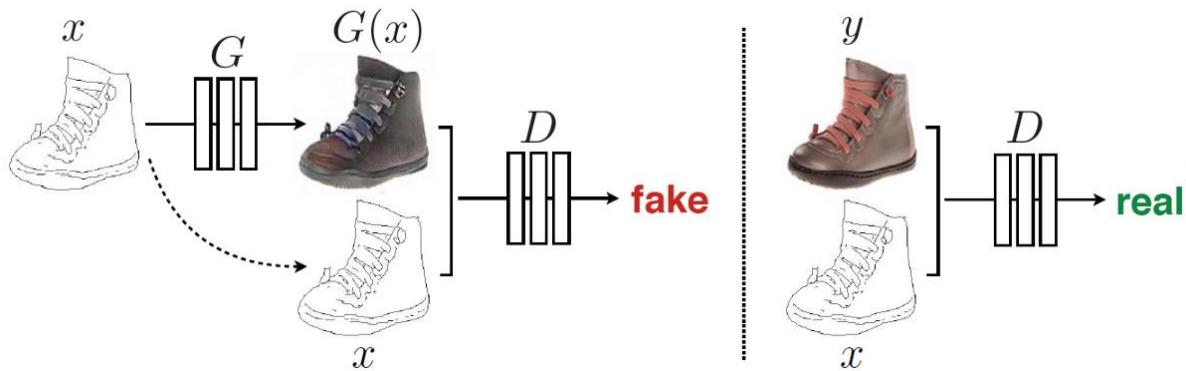
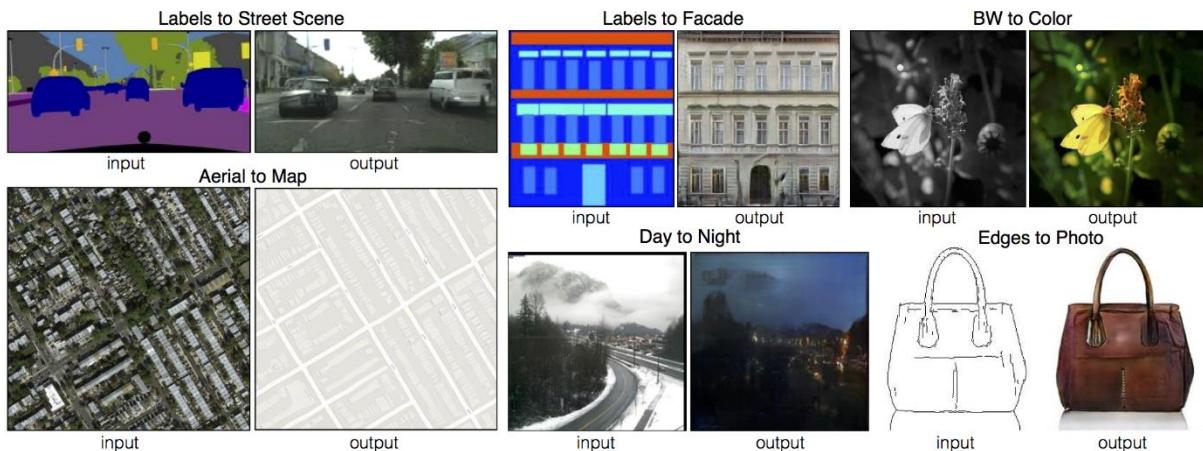


Image-To-Image Translation is the process for translating one representation of the image into another representation.

The image-to-image translation is another example of a task that GANs (Generative Adversarial Networks) are ideally suited for. These are tasks in which it is nearly impossible to hard-code a loss function. Studies on GANs are concerned with novel image synthesis, translating from a random vector z into an image. Image-to-Image translation converts one image to another like the edges of the bag below to the photo image. Another exciting example of this is shown below:



In Pix2Pix Dual Objective Function with an Adversarial and L1 Loss

A naive way to do Image-to-Image translation would be to discard the adversarial framework altogether. A source image would just be passed through a parametric function, and the difference in the resulting image and the ground truth output would be used to update the weights of the network. However, designing this loss function with standard distance measures such as L1 and L2 will fail to capture many of the essential distinctive characteristics between these images. However,

authors do find some value to the L1 loss function as a weighted sidekick to the adversarial loss function.

The Conditional-Adversarial Loss (Generator versus Discriminator) is very popularly formatted as follows:

$$\begin{aligned}\mathcal{L}_{cGAN}(G, D) = & \mathbb{E}_{x,y}[\log D(x, y)] + \\ & \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))]\end{aligned}$$

The L1 loss function previously mentioned is shown below:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1].$$

Combining these functions results in:

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G).$$

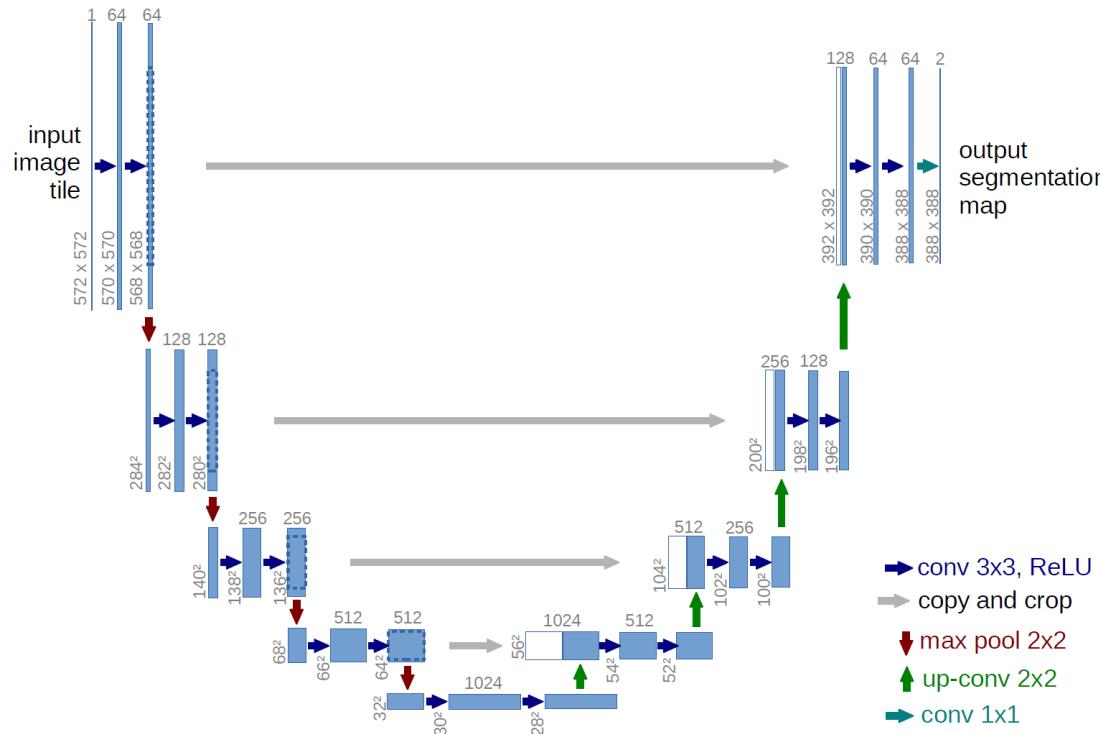
In the experiments, the authors report that they found the most success with the lambda parameter equal to 100.

Q4. Explain UNet Architecture?

Answer:

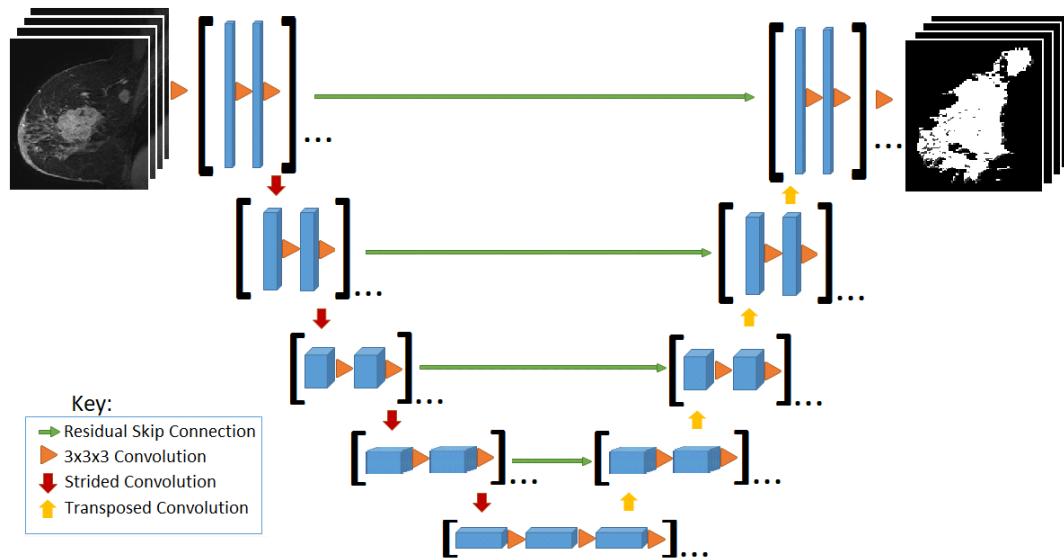
U-Net architecture: It is built upon the Fully Convolutional Network and modified in a way that it yields better segmentation in medical imaging. Compared to FCN-8, the two main differences are (a) U-net is symmetric and (b) the skip connections between the downsampling path and upsampling path apply a concatenation operator instead of a sum. These skip connections intend to provide local information to the global information while upsampling. Because of its symmetry, the network has a large number of feature maps in the upsampling path, which allows transferring information. By comparison, the underlying FCN architecture only had the *number of classes* feature maps in its upsampling way.

How does it work?



The UNet architecture looks like a ‘U,’ which justifies its name. This UNet architecture consists of 3 sections: The contraction, the bottleneck, and the expansion section. The contraction section is made of many contraction blocks. Each block takes an input that applies two 3×3 convolution layers, followed by a 2×2 max pooling. The number of features or kernel maps after each block doubles so that UNet architecture can learn complex structures. Bottommost layer mediates between the contraction layer and the expansion layer. It uses two 3×3 CNN layers followed by 2×2 up convolution layer.

But the heart of this architecture lies in the expansion section. Similar to the contraction layer, it also has several expansion blocks. Each block passes input to two 3×3 CNN layers, followed by a 2×2 upsampling layer. After each block number of feature maps used by the convolutional layer, get half to maintain symmetry. However, every time input is also get appended by feature maps of the corresponding contraction layer. This action would ensure that features that are learned while contracting the image will be used to reconstruct it. The number of expansion blocks is as same as the number of contraction blocks. After that, the resultant mapping passes through another 3×3 CNN layer, with the number of feature maps equal to the number of segments desired.



Q5. What is pair2vec?

Answer:

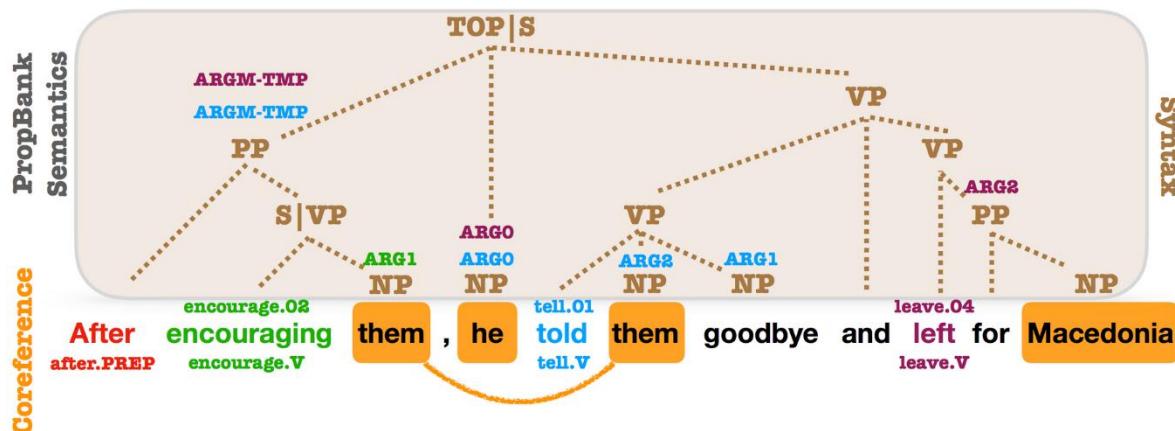
This paper pre trains *word pair representations* by maximizing pointwise mutual information of pairs of words with their context. This encourages a model to learn more meaningful representations of word pairs than with more general objectives, like modeling. The pre-trained representations are useful in tasks like SQuAD and MultiNLI that require cross-sentence inference. You can expect to see more pretraining tasks that capture properties particularly suited to specific downstream tasks and are complementary to more general-purpose tasks like language modeling.

Reasoning about implied relationships between pairs of words is crucial for cross sentences inference problems like question answering (QA) and natural language inference (NLI). In NLI, e.g., given a premise such as “*golf is prohibitively expensive*,” inferring that the hypothesis “*golf is a cheap pastime*” is a contradiction requires one to know that *expensive* and *cheap* are antonyms. Recent work has shown that current models, which rely heavily on unsupervised single-word embeddings, struggle to grasp such relationships. In this pair2vec paper, we show that they can be learned with word pair2vec(pair vector), which are trained, unsupervised, at a huge scale, and which significantly improve performance when added to existing cross-sentence attention mechanisms.

X	Y	Contexts
		with X and Y baths
hot	cold	too X or too Y neither X nor Y in X, Y
Portland	Oregon	the X metropolitan area in Y X International Airport in Y
crop	wheat	food X are maize, Y, etc dry X, such as Y, more X circles appeared in Y fields
Android	Google	X OS comes with Y play the X team at Y X is developed by Y

Table 1: Example word pairs (italicized) and their contexts (Wikipedia).

Unlike single word representations, which are typically trained by modeling the co-occurrence of a target word x with its context c , our word-pair representations are learned by modeling the three-way co-occurrence between two words (x,y) and the context c that ties them together, as illustrated in above Table. While similar training signal has been used to learn models for ontology construction and knowledge base completion, this paper shows, for the first time, that considerable scale learning of pairwise embeddings can be used to improve the performance of neural cross-sentence inference models directly.



Q6. What is Meta-Learning?

Answer:

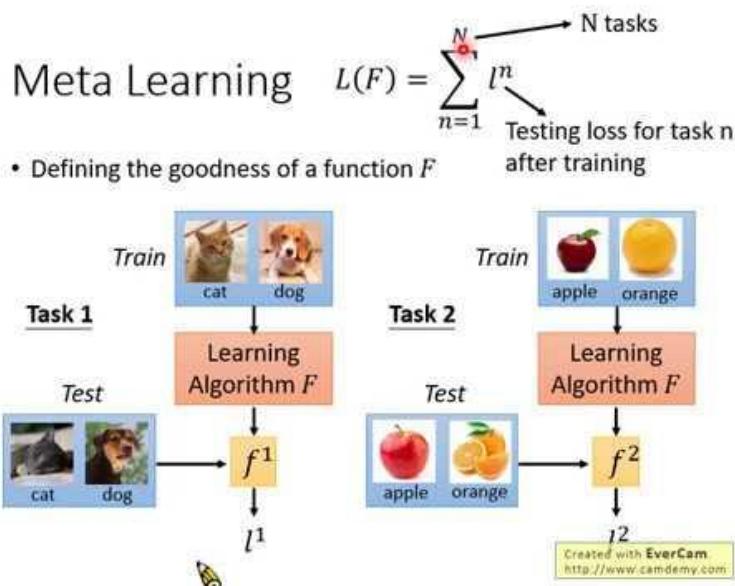
Meta-learning: It is an exciting area of research that tackles the problem of learning to learn. The goal is to design models that can learn new skills or fastly to adapt to new environments with minimum training examples. Not only does this dramatically speed up and improve the design of ML(Machine Learning) pipelines or neural architectures, but it also allows us to replace hand-engineered algorithms with novel approaches learned in a data-driven way.

The goal of meta-learning is to train the model on a variety of learning tasks, such that it can solve new learning tasks with only a small number of training samples. It tends to focus on finding **model agnostic** solutions, whereas multi-task learning remains deeply tied to model architecture.

Thus, meta-level AI algorithms make AI systems:

- Learn faster
- Generalizable to many tasks
- Adaptable to environmental changes like in Reinforcement Learning

One can solve any problem with a single model, but meta-learning should not be confused with one-shot learning.



Q7. What is ALiPy(Active Learning in Python)?

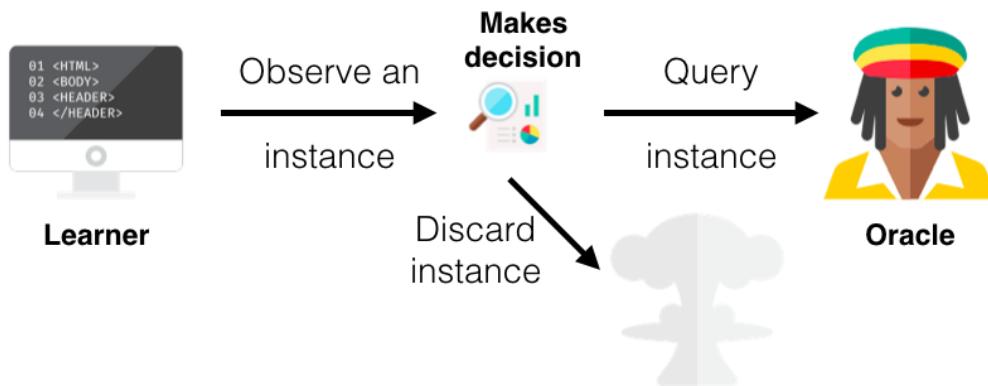
Answer:

Supervised ML methods usually require a large set of labeled examples for model training. However, in many real applications, there are ample unlabeled data but limited labeled data; and acquisition of labels is costly. Active learning (AL) reduces labeling costs by iteratively selecting the most valuable data to query their labels from the annotator.

Active learning is the leading approach to learning with limited labeled data. It tries to reduce human efforts on data annotation by actively querying the most prominent examples.

ALiPy is a Python toolbox for active learning(AL), which is suitable for various users. On the one hand, the entire process of active learning has been well implemented. Users can efficiently perform experiments by many lines of codes to finish the entire process from data pre-processes to

result in visualization. More than 20 commonly used active learning(AL) methods have been implemented in the toolbox, providing users many choices.



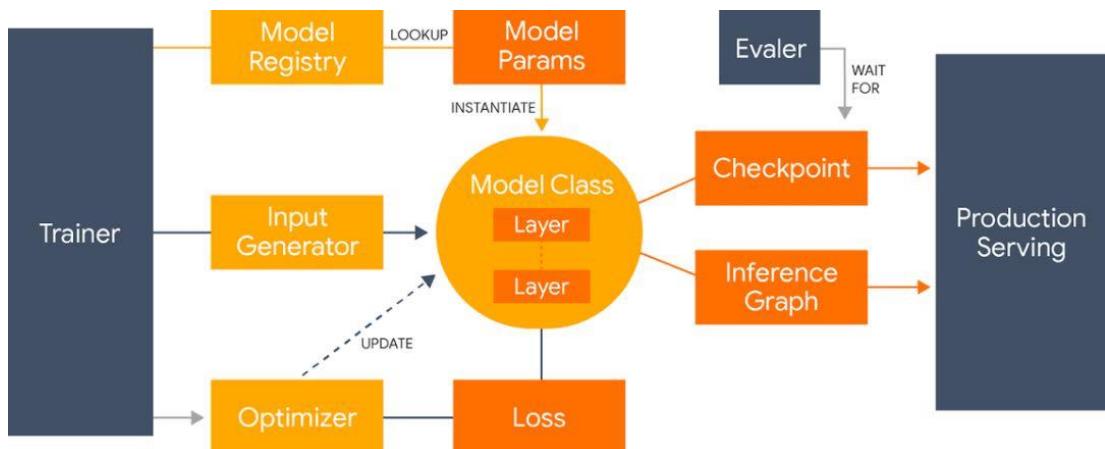
Q8.What is the Lingvo model?

Answer:

Lingvo: It is a Tensorflow framework offering a complete solution for collaborative deep learning research, with a particular focus towards sequence-to-sequence models. These models are composed of modular building blocks that are flexible and easily extensible, and experiment configurations are centralized and highly customizable. Distributed training and quantized inference are supported directly within a framework, and it contains existing implementations of an ample number of utilities, helper functions, and newest research ideas. This model has been used in collaboration by dozens of researchers in more than 20 papers over the last two years.

Why does this Lingvo research matter?

The process of establishing a new deep learning(DL) system is quite complicated. It involves exploring an ample space of design choices involving training data, data processing logic, the size, and type of model components, the optimization procedures, and the path to deployment. This complexity requires the framework that quickly facilitates the production of new combinations and the modifications from existing documents and experiments and shares these new results. It is a workspace ready to be used by deep learning researchers or developers. Nguyen Says: “We have researchers working on state-of-the-art(SOTA) products and research algorithms, basing their research off of the same codebase. This ensures that code is battle-tested. Our collective experience is encoded in means of good defaults and primitives that we have found useful over these tasks.”



Q9. What is Dropout Neural Networks?

Answer:

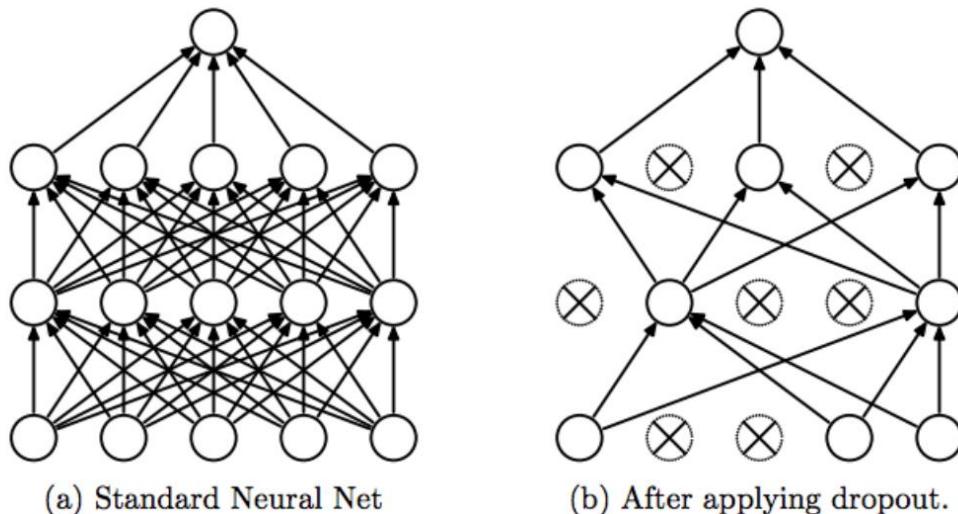
The term “dropout” refers to dropping out units (both hidden and visible) in a neural network.

At each training stage, individual nodes are either dropped out of the net with probability $1-p$ or kept with probability p , so that a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed.

Why do we need Dropout?

The answer to these questions is “to prevent over-fitting.”

A fully connected layer occupies most of the parameters, and hence, neurons develop co-dependency amongst each other during training, which curbs the individual power of each neuron leading to over-fitting of training data.

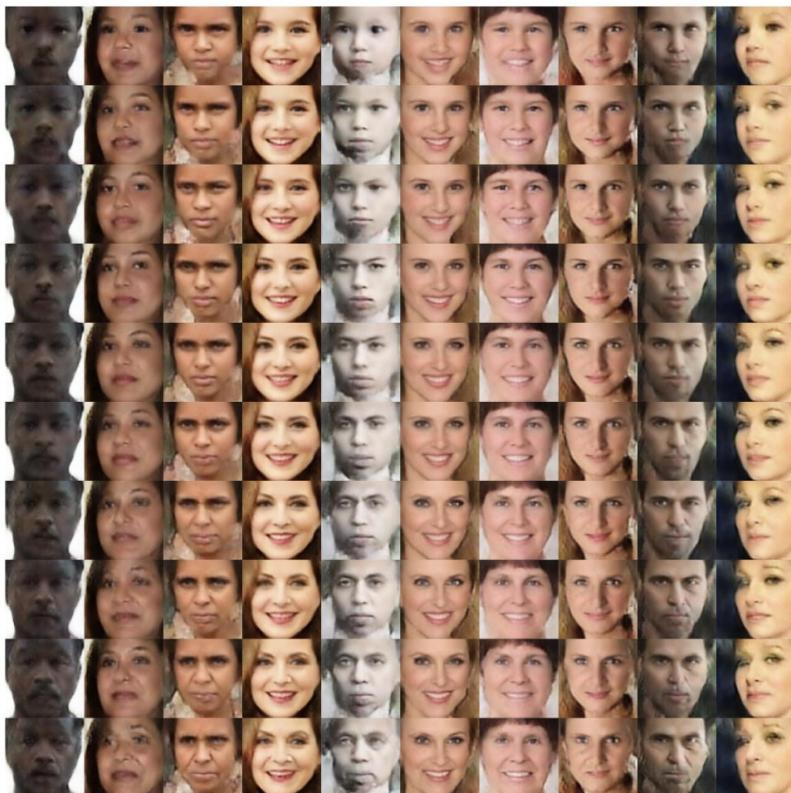


Q10. What is GAN?

Answer:

A generative adversarial network (GAN): It is a class of machine learning systems invented by Ian Goodfellow and his colleagues in 2014. Two neural networks are contesting with each other in a game (in the idea of game theory, often but not always in the form of a zero-sum game). Given a training set, this technique learns to generate new data with the same statistics as the training set. E.g., a GAN trained on photographs can produce original pictures that look at least superficially authentic to human observers, having many realistic characteristics. Though initially proposed as a form of a generative model for unsupervised learning, GANs have also proven useful for semi-supervised learning,^[2] fully supervised learning, and reinforcement learning.

Example of GAN



- Given an image of a face, the network can construct an image that represents how that person could look when they are old.

Generative Adversarial Networks takes up a game-theoretic approach, unlike a conventional neural network. The network learns to generate from a training distribution through a 2-player game. The two entities are Generator and Discriminator. These two adversaries are in constant battle throughout the training process.

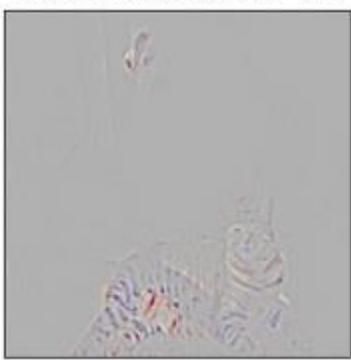
Q1. Explain Grad-CAM architecture?

Answer:

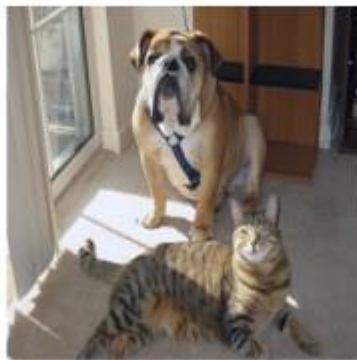
According to the research paper, “We propose a technique for making Convolutional Neural Network (CNN)-based models more transparent by visualizing input regions that are ‘important’ for predictions – producing *visual explanations*. Our approach is called Gradient-weighted Class Activation Mapping (Grad-CAM), which uses class-specific gradient information to localize the crucial regions. These localizations are combined with the existing pixel-space visualizations to create a new high-resolution, and class-discriminative display called the Guided Grad-CAM. These methods help better to understand CNN-based models, including image captioning and the apparent question answering (VQA) models. We evaluate our visual explanations by measuring the ability to discriminate between the classes and to inspire trust in humans, and their correlation with the occlusion maps. Grad-CAM provides a new way to understand the CNN-based models.”

A technique for making CNN(Convolutional Neural Network)-based models more transparent by visualizing the regions of input that are “important” for predictions from these models — or visual explanations.

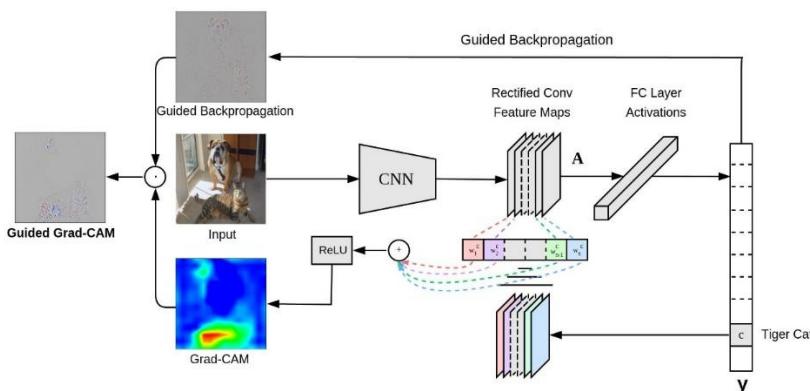
Guided Grad-CAM for “Cat”



Guided Grad-CAM for “Dog”



This visualization is both high-resolution (when the class of interest is ‘tiger cat,’ it identifies crucial ‘tiger cat’ features like stripes, pointy ears and eyes) and class-discriminative (it shows the ‘tiger cat’ but not the ‘boxer (dog)’).



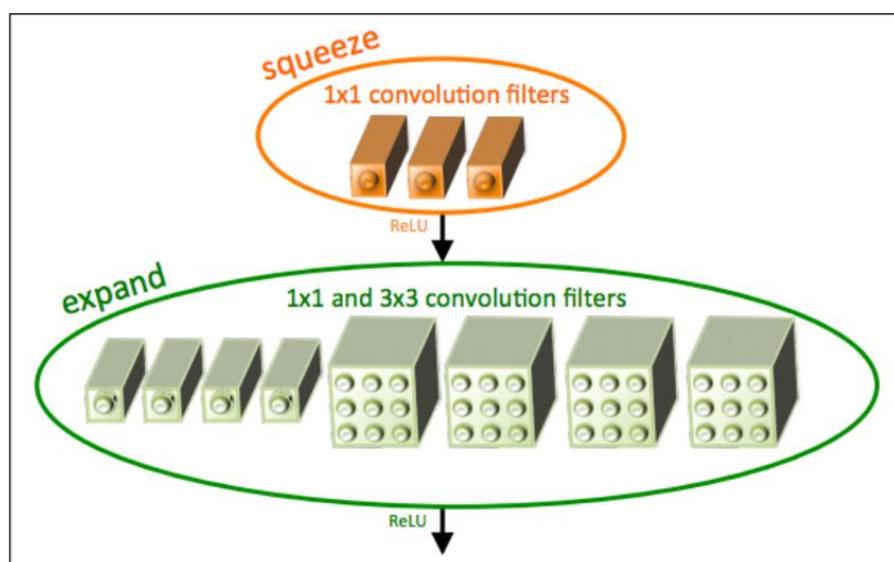
Q2.Explain squeeze-net architecture?

Answer:

Nowadays, technology is at its peak. Self-driving cars and IoT is going to be household talks in the next few years to come. Therefore, everything is controlled remotely, say, e.g., in self-driving cars, we will need our system to communicate with the servers regularly. So accordingly, if we have a model that has a small size, then we can quickly deploy it in the cloud. So that's why we needed an architecture that is less in size and also achieves the same level of accuracy that other architecture achieves.

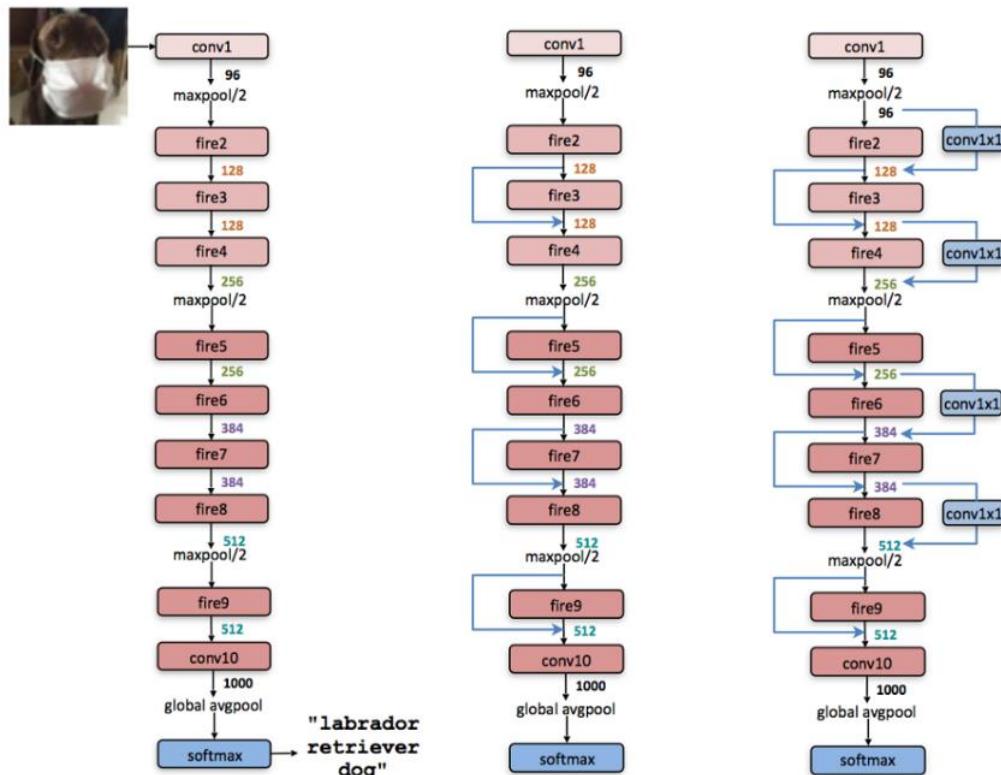
It's Architecture

- **Replace 3x3 filters with 1x1 filter-** We plan to use the maximum number of 1x1 filters as using a 1X1 filter rather than a 3X3 filter can reduce the number of parameters by 9X. We may think that replacing 3X3 filters with 1X1 filters may perform badly as it has less information to work on. But this is not a case. Typically 3X3 filter may capture the spatial information of pixels close to each other while the 1X1 filter zeros in on pixel and captures features amongst its channels.
- **Decrease number of input channels to 3x3 filters-** to maintain a small total number of parameters in a CNN, and it is crucial not only to decrease the number of 3x3 filters, but also to decrease the number of input channels to 3x3 filters. We reduce the number of input channels to 3x3 filters using *squeeze layers*. The author of this paper has used a term called the “*fire module*,” in which there is a squeeze layer and an expanded layer. In the squeeze layer, we are using 1X1 filters, while in the expanded layer, we are using a combo of 3X3 filters and 1X1 filters. The author is trying to limit the number of inputs to 3X3 filters to reduce the number of parameters in the layer.



- **Downsample late in a network so that convolution layers have a large activation map-** Having got an intuition about contracting the sheer number of parameters we are working with, how the model is getting most out of the remaining set of parameters. The author in this paper has downsampled the feature map in later layers, and this increases the accuracy. But this is an excellent contrast to networks like VGG where a large feature map is taken, and then it gets smaller as network approach towards the end. This different approach is too interesting, and they cite the [paper by K. He and H. Sun](#) that similarly applies delayed downsampling that leads to higher classification accuracy.

This architecture consists of the fire module, which enables it to bring down the number of parameters.



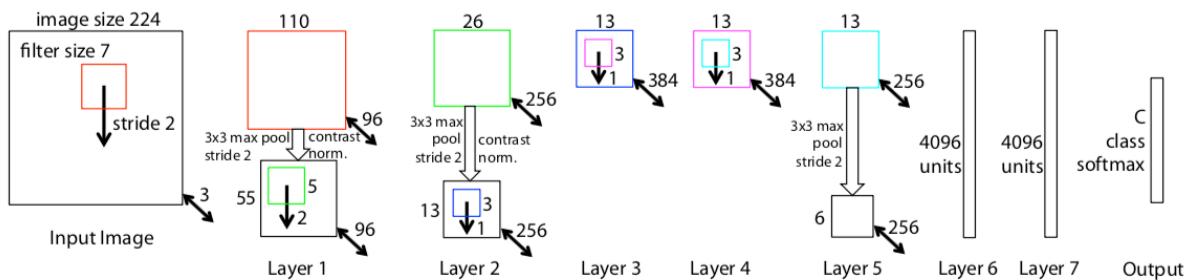
And other thing that surprises me is the lack of fully connected layers or dense layers at the end, which one will see in a typical CNN architecture. The dense layers, in the end, learn all the relationships between the high-level features and the classes it is trying to identify. The fully connected layers are designed to learn that noses and ears make up a face, and wheels and lights indicate cars. However, in this architecture, that extra learning step seems to be embedded within the transformations between various “fire modules.”

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD [5]	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning [11]	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression [10]	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	50x	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	363x	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	510x	57.5%	80.3%

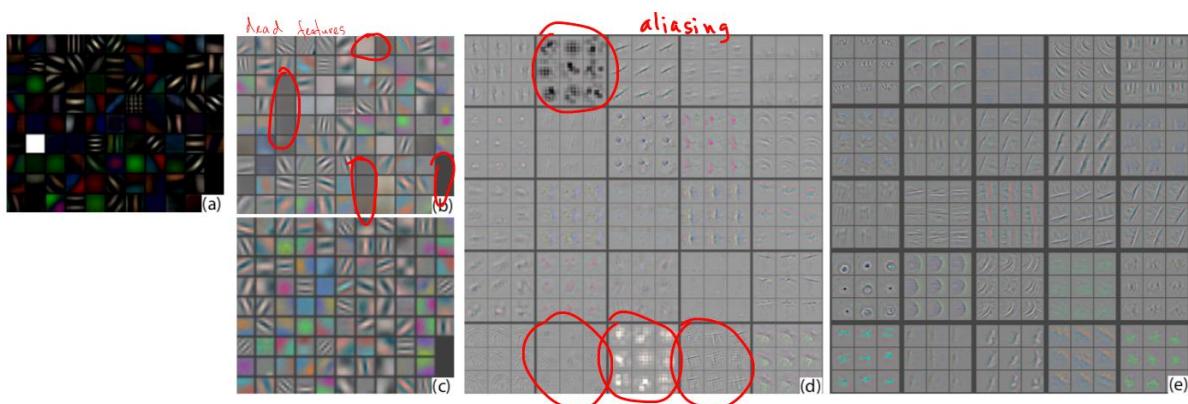
The squeeze-net can accomplish an accuracy nearly equal to AlexNet with 50X less number of parameters. The most impressive part is that if we apply Deep compression to the already smaller model, then it can reduce the size of the squeeze-net model to 510x times that of AlexNet.

Q3.ZFNet architecture

Answer:



The architecture of the network is an optimized version of the last year's winner - AlexNet. The authors spent some time to find out the bottlenecks of AlexNet and removing them, achieving superior performance.



(a): First layer ZFNET features without feature scale clipping. (b): the First layer features from AlexNet. Note that there are lot of dead features - ones where the network did not learn any patterns. (c): the First layer features for ZFNet. Note that there are only a few dead features. (d): Second layer features from AlexNet. The grid-like patterns are so-called aliasing artifacts. They appear when

receptive fields of convolutional neurons overlap, and neighboring neurons learn similar structures. (e): 2nd layer features for ZFNet. Note that there are no aliasing artifacts. Source: original paper.

In particular, they reduced the filter size in the 1st convolutional layer from 11x11 to 7x7, which resulted in fewer dead features learned in the first layer (see the image below for an example of that). A dead feature is a situation where a convolutional kernel fails to learn any significant representation. Visually it looks like a monotonic single-color image, where all the values are close to each other.

In addition to changing the filter size, the authors of FZNet have doubled the number of filters in all convolutional layers and the number of neurons in the fully connected layers as compared to the AlexNet. In the AlexNet, there were 48-128-192-192-128-2048-2048 kernels/neurons, and in the ZFNet, all these doubled to 96-256-384-384-256-4096-4096. This modification allowed the network to increase the complexity of internal representations and as a result, decrease the error rate from 15.4% for last year's winner, to 14.8% to become the winner in 2013.

Q4. What is NAS (Neural Architecture Search)?

Answer:

Developing the neural network models often requires significant architecture engineering. We can sometimes get by with transfer learning, but if we want the best possible performance, it's usually best to design your network. This requires specialized skills and is challenging in general; we may not even know the limits of the current state-of-the-art(SOTA) techniques. Its a lot of trial and error, and experimentation itself is time-consuming and expensive.

This is the NAS(Neural Architecture Search) comes in. NAS(Neural Architecture Search) is an algorithm that searches for the best neural network architecture. Most of the algorithms work in the following way. Start off by defining the set of “building blocks” that can be used for our network. E.g., the state-of-the-art(SOTA) NASNet paper proposes these commonly used blocks for an image recognition network-

- identity
- 1x7 then 7x1 convolution
- 3x3 average pooling
- 5x5 max pooling
- 1x1 convolution
- 3x3 depthwise-separable conv
- 7x7 depthwise-separable conv
- 1x3 then 3x1 convolution
- 3x3 dilated convolution
- 3x3 max pooling
- 7x7 max pooling
- 3x3 convolution
- 5x5 depthwise-separable conv

In the NAS algorithm, the controller Recurrent Neural Network (RNN) samples the building blocks, putting them together to create some end to end architecture. Architecture generally combines the same style as state-of-the-art(SOTA) networks, such as DenseNets or ResNets, but uses a much different combination and the configuration of blocks.

This new network architecture is then trained to convergence to obtain the least accuracy on the held-out validation set. The resulting efficiencies are used to update the controller so that the controller will generate better architectures over time, perhaps by selecting better blocks or making better connections. The controller weights are updated with a policy gradient. The whole end-to-end setup is shown below.

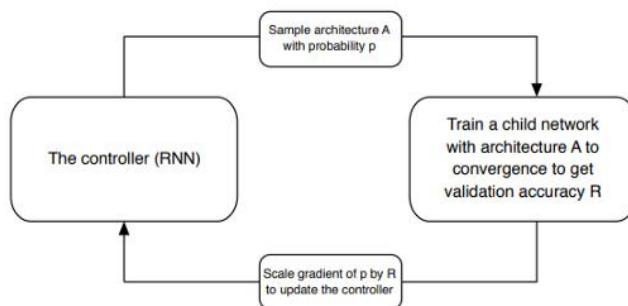
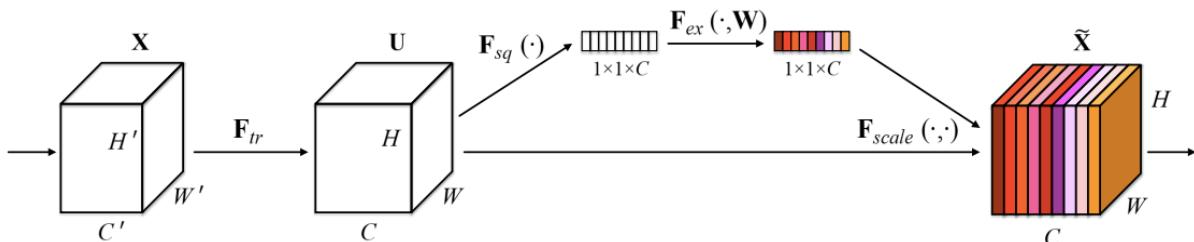


Figure 1. Overview of Neural Architecture Search [71]. A controller RNN predicts architecture A from a search space with probability p . A child network with architecture A is trained to convergence achieving accuracy R . Scale the gradients of p by R to update the RNN controller.

It's a reasonably intuitive approach! In simple means: have an algorithm grab different blocks and put those blocks together to make the network. Train and test out that network. Based on our results, adjust the blocks we used to make the network and how you put them together!

Q5. What is SENets?

Answer:



SENet stands for Squeeze-and-Excitation Networks introduces a building block for CNNs that improves channel interdependencies at almost no computational cost. They have used in the 2017 ImageNet competition and helped to improve the result from last year by 25%. Besides this large performance boost, they can be easily added to existing architectures. The idea is this:

Let's add parameters to each channel of the convolutional block so that the network can adaptively adjust the weighting of each feature map.

As simple as may it sound, this is it. So, let's take a closer look at why this works so well.

Why it works too well?

CNN's uses its convolutional filters to extract hierachal information from the images. Lower layers find little pieces of context like high frequencies or edges, while upper layers can detect faces, text, or other complex geometrical shapes. They extract whatever is necessary to solve the task precisely.

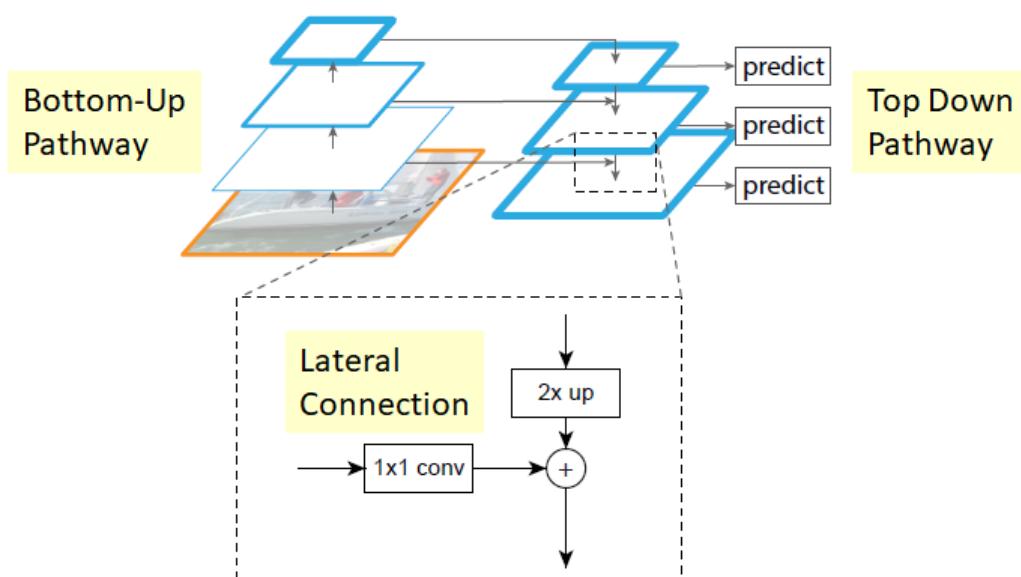
All of this works by fusing spatial and channel information of an image. The different filters will first find the spatial features in each input channel before adding the information across all available output channels.

All we need to understand for now is that the network weights each of its channels equally when creating output feature maps. It is all about changing this by adding a content-aware mechanism to weight each channel adaptively. In its too basic form, this could mean adding a single parameter to each channel and giving it linear scalar how relevant each one is.

However, the authors push it a little further. First, they get the global understanding of each channel by squeezing feature maps to a single numeric value. This results in the vector of size n , where n is equal to the number of convolutional channels. Afterward, it is fed through a two-layer neural network, which outputs a vector of the same size. These n values can now be used as weights on the original features maps, scaling each channel based on its importance.

Q6. Feature Pyramid Network (FPN)

Answer:



The Bottom-Up Pathway

The bottom-up pathway is feedforward computation of backbone ConvNet. It is known as one pyramid level is for each stage. The output of last layer of each step will be used as the reference set of feature maps for enriching the top-down pathway by lateral connection.

Top-Down Pathway and Lateral Connection

- The higher resolution features are upsampled spatially coarser, but semantically stronger, feature maps from higher pyramid levels. More particularly, the spatial resolution is upsampled by a factor of 2 using nearest neighbor for simplicity.
- Each lateral connection adds feature maps of the same spatial size from the bottom-up pathway and top-down pathway.
- Specifically, **the feature maps from the bottom-up pathway undergo 1×1 convolutions to reduce channel dimensions.**
- And **feature maps from the bottom-up pathway and top-down pathway are merged by element-wise addition.**

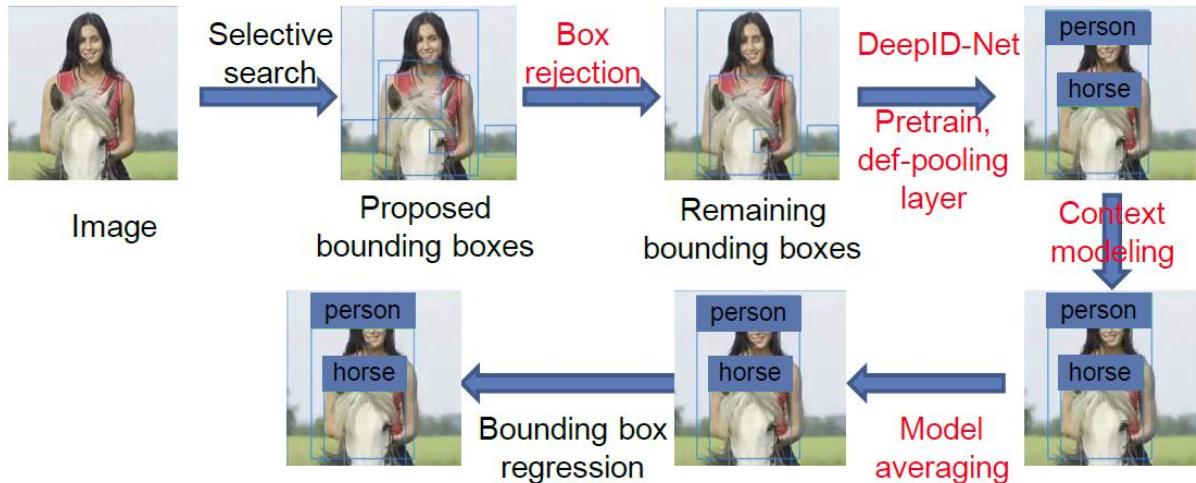
Prediction in FPN

- Finally, **the 3×3 convolution is appended on each merged map to generate a final feature map, which is to reduce the aliasing effect of upsampling.** This last set of feature maps is called $\{P_2, P_3, P_4, P_5\}$, corresponding to $\{C_2, C_3, C_4, C_5\}$ that are respectively of same spatial sizes.
- Because all levels of pyramid use shared classifiers/regressors as in a traditional featured image pyramid, feature dimension at output d is fixed with $d = 256$. Thus, all extra convolutional layers have 256 channel outputs.

Q7. DeepID-Net(Def-Pooling Layer)

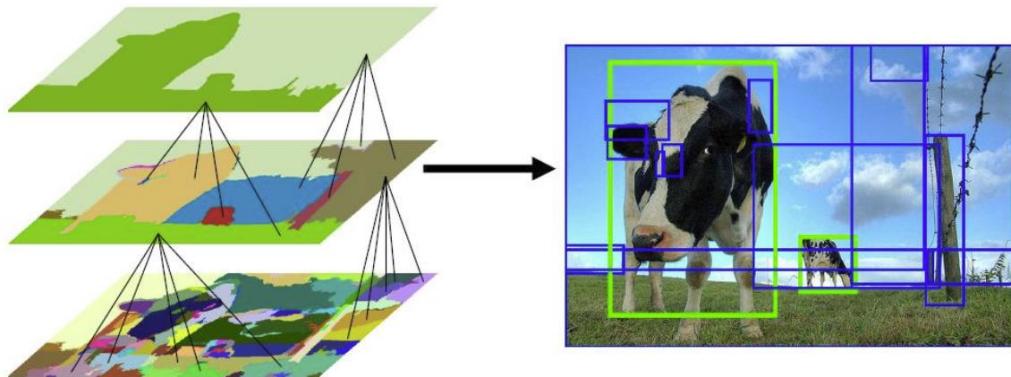
Answer:

A new def-pooling (deformable constrained pooling) layer is used to model the deformation of the object parts with geometric constraints and penalties. That means, except detecting the whole object directly, it is also important to identify object parts, which can then assist in detecting the whole object.



The steps in **black** color are the **old stuff** that existed in R-CNN. The stages in **red** color do not appear in R-CNN.

1. Selective Search

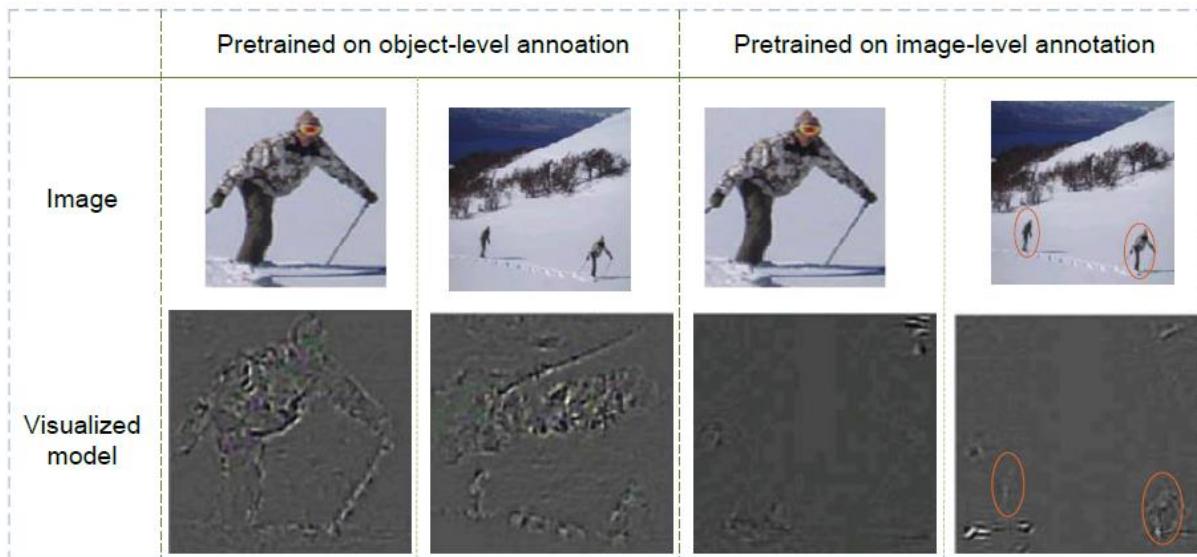


- First, color similarities, texture similarities, regions size, and region filling are used as **non-object-based segmentation**. Therefore you obtain **many small segmented areas** as shown at the bottom left of the image above.
- Then, the bottom-up approach is used that **small segmented areas are merged to form the larger segment areas**.
- Thus, **about 2K regions, proposals (bounding box candidates) are generated**, as shown in the above image.

2. Box Rejection

R-CNN is used to reject bounding boxes that are most likely to be the background.

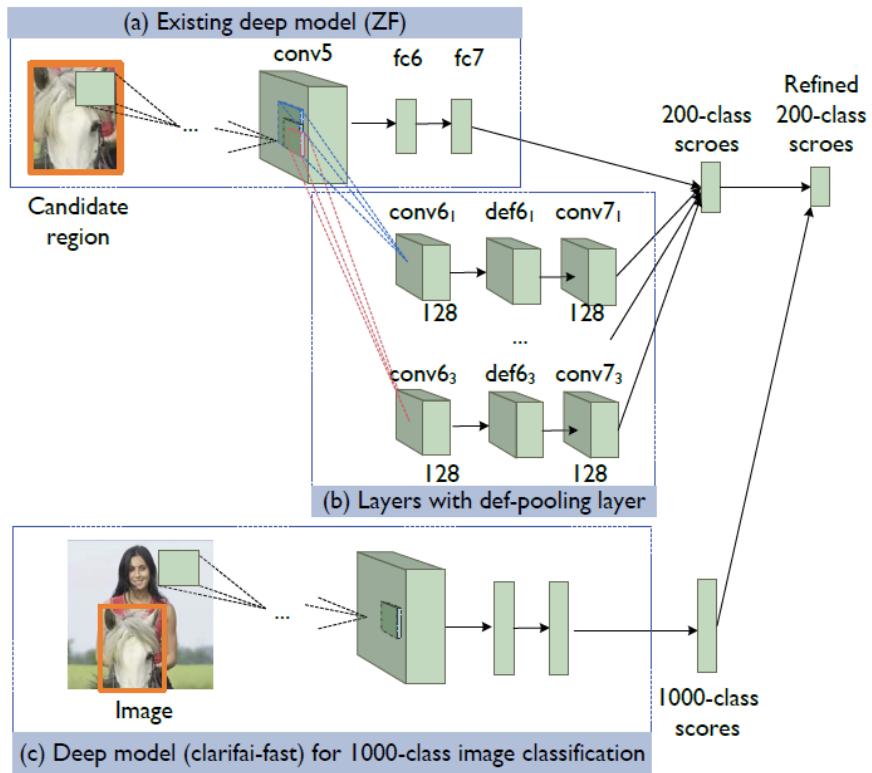
3. Pre train Using Object-Level Annotations

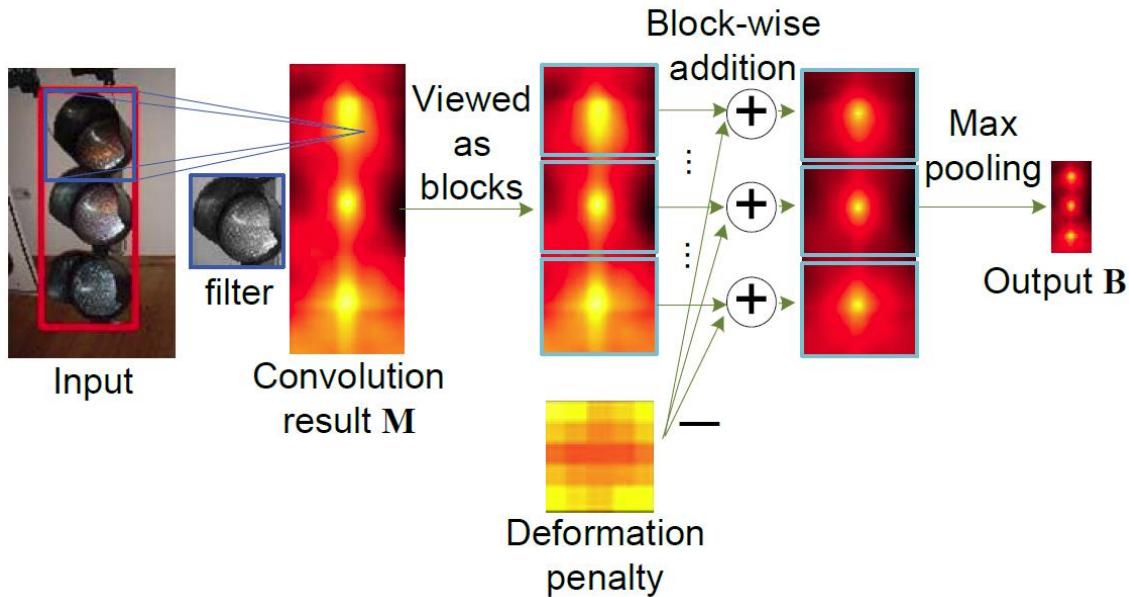


Usually, pretraining is on **image-level annotation**. It is **not good when an object is too small within the image** because the object should occupy a large area within the bounding box created by the selective search.

Thus, **pretraining is on object-level annotation**. And **the deep learning(DL) model can be any models** such as ZFNet, VGGNet, and GoogLeNet.

4. Def-Pooling Layer





$$b_c^{(x,y)} = \max_{\delta_x, \delta_y \in \{-R, \dots, R\}} \{ m_c^{\mathbf{z}_{\delta_x, \delta_y}} - \sum_{n=1}^N a_{c,n} d_{c,n}^{\delta_x, \delta_y} \}$$

where $\mathbf{z}_{\delta_x, \delta_y} = (s_x \cdot x + \delta_x, s_y \cdot y + \delta_y)$.

For the def-pooling path, output from conv5, goes through the Conv layer, then goes through the def-pooling layer, and then has a max-pooling layer.

In simple terms, the summation of ac multiplied by dc,n, is the 5×5 deformation penalty in the figure above. The penalty of placing object part from assumed the central position.

By training the DeepID-Net, object parts of the object to be detected will give a high activation value after the def-pooling layer if they are closed to their anchor places. And this output will connect to 200-class scores for improvement.

5. Context Modeling

In object detection tasks in ILSVRC, there are 200 classes. And there is also the classification competition task in ILSVRC for classifying and localizing 1000-class objects. The contents are more diverse compared with the object detection task. Hence, **1000-class scores, obtained by classification network, are used to refine 200-class scores**.

6. The Model Averaging-

Multiple models are used to increase the accuracy, and **the results from all models are averaged**. This technique has been used since AlexNet, LeNet, and so on.

7. Bounding Box Regression

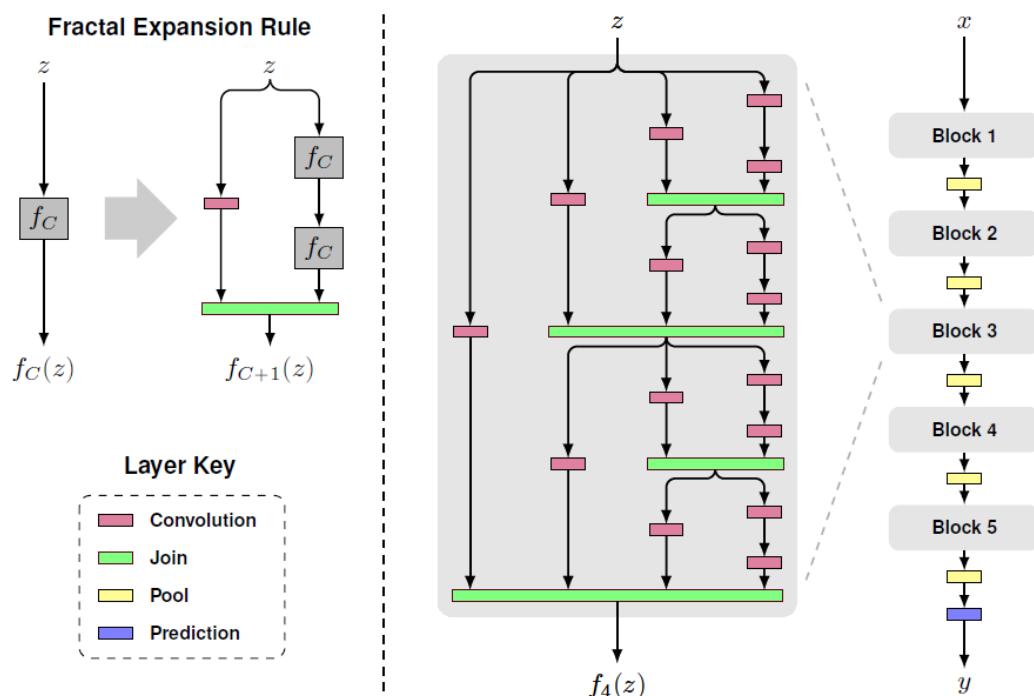
Bounding box regression is to **fine-tune the bounding box location**, which has been used in R-CNN.

Q8. What is FractalNet Architecture?

Answer:

In 2015, after the invention of ResNet, with numerous champion won, there are plenty of researchers working on how to improve the ResNet, such as Pre-Activation ResNet, RiR, RoR, Stochastic Depth, and WRN. In this story, conversely, a non-residual-network approach, FractalNet, is shortly reviewed. When VGGNet is starting to degrade when it goes from 16 layers (VGG-16) to 19 layers (VGG-19), FractalNet can go up to 40 layers or even 80 layers.

Architecture



In the above picture: A Simple Fractal Expansion (on Left), Recursively Stacking of Fractal Expansion as One Block (in the Middle), 5 Blocks Cascaded as FractalNet (on the Right)

For the base case, $f_1(z)$ is the convolutional layer:

$$f_1(z) = \text{conv}(z)$$

After that, recursive fractals are:

$$f_{C+1}(z) = [(f_C \circ f_C)(z)] \oplus [\text{conv}(z)]$$

Where C is a number of columns as in the middle of the above figure. The number of the convolutional layers at the deepest path within the block will have $2^{(C-1)}$. In this case, $C=4$, thereby, a number of convolutional layers are $2^3=8$ layers.

For the **join layer** (green), the **element-wise mean** is computed. It is not concatenation or addition.

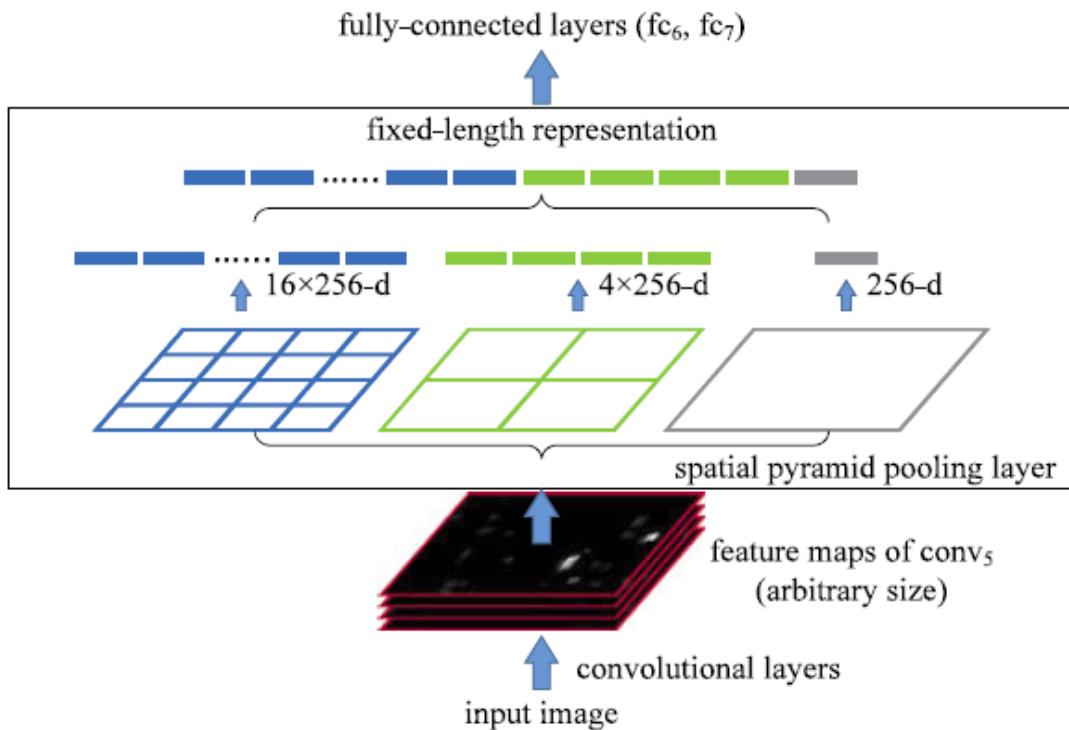
With five blocks ($B=5$) cascaded as FractalNet at the right of the figure, then the number of convolutional layers at the most profound path within the whole network is $B \times 2^{(C-1)}$, i.e., $5 \times 2^3 = 40$ layers.

In between 2 blocks, 2×2 max pooling is done to reduce the size of feature maps. Batch Norm and ReLU are used after each convolution.

Q9. What is the SSPNet architecture?

Answer:

SPPNet has introduced the new technique in CNN called **Spatial Pyramid Pooling (SPP)** at the transition of the convolutional layer and fully connected layer. This is a work from **Microsoft**.



Conventionally, at the transformation of the Conv layer and FC layer, there is one single pooling layer or even no pooling layer. In SPPNet, it suggests having **multiple pooling layers with different scales**.

In the figure, **3-level SPP** is used. Suppose conv5 layer has 256 feature maps. Then at the SPP layer,

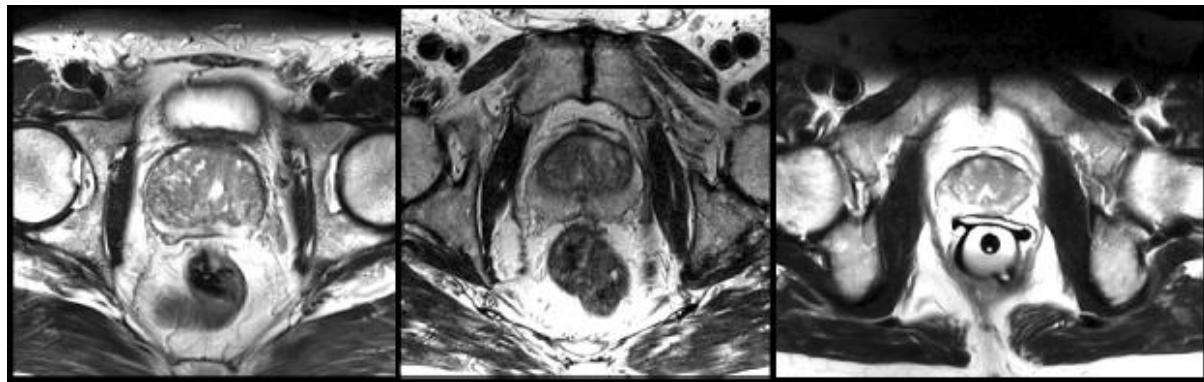
-
1. first, each feature map is **pooled to become one value (which is grey)**. Thus **256-d vector is formed**.
 2. Then, each feature map is **pooled to have four values (which is green)**, and form the **4×256-d vector**.
 3. Similarly, each feature map is **pooled to have 16 values (in blue)**, and form the **16×256-d vector**.
 4. The **above three vectors are concatenated to form a 1-d vector**.
 5. Finally, this **1-d vector is going into FC layers** as usual.

With SPP, you don't need to crop the image to a fixed size, like AlexNet, before going into CNN. **Any image sizes can be inputted.**

Q1. Explain V-Net (Volumetric Convolution) Architecture with related to Biomedical Image Segmentation?

Answer:

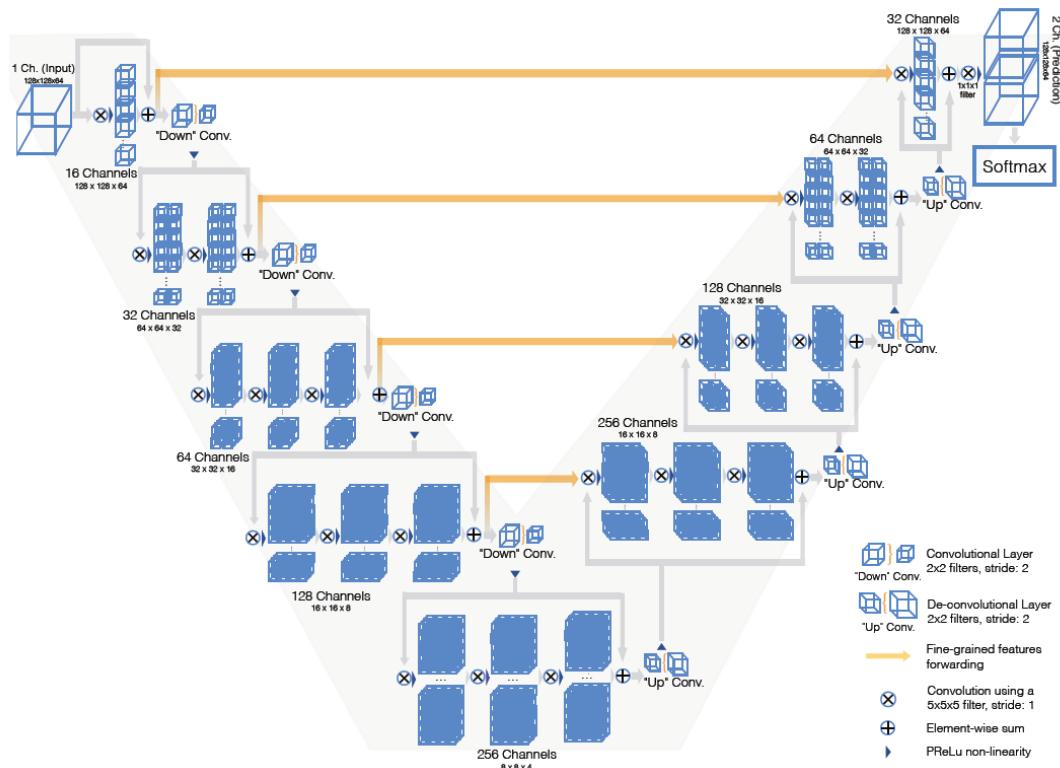
There were several medical data used in clinical practice consists of 3D volumes, such as MRI volumes illustrate prostate, while most approaches are only able to process 2D images. A 3D image segmentation based on a volumetric, fully convolutional neural network is proposed in this work.



Slices from MRI volumes depicting prostate

Prostate segmentation nevertheless is the crucial task having clinical relevance both during diagnosis, where the volume of the prostate needs to be assessed and during treatment planning, where the estimate of the anatomical boundary needs to be accurate.

Architecture



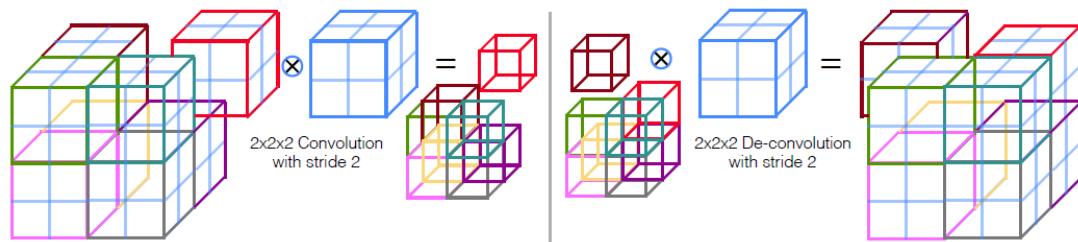
- V-Net, justifies by its name, it is shown as V-shape. The left part of the network consists of a compression path, while on the right part decompresses signal until its original size is reached.
- This is the same as U-Net, but with some difference.

On Left

- The left side of the network is divided into different stages that operate at various resolutions. Each stage comprises one to 3 convolutional layers.
- **At each stage, a residual function is learned.** The input of each stage is used in convolutional layers and processed through non-linearities and added to the output of the last convolutional layer of that stage to enable learning a residual function. This V-net architecture ensures convergence compared with non-residual learning networks such as U-Net.
- The **convolutions** performed in each stage use **volumetric kernels** having the size of **5×5×5 voxels**. (A voxel represents a value on a regular grid in 3D-space. The term voxel is commonly used in 3D much 3D space, just like voxelization in a point cloud.)
- Along the compression path, the **resolution is reduced by convolution with 2×2×2 voxels full kernels applied with stride 2**. Thus, the size of the resulting feature maps is halved, with a **similar purpose as pooling layers**. And **number of feature channels doubles at each stage** of the compression path of V-Net.
- Replacing pooling operations with convolutional ones helps to have a smaller memory footprint during training because no switches mapping the output of pooling layers back to their inputs are needed for back-propagation.
- Downsampling helps to increase the receptive field.
- **PReLU** is used as a non-linearity activation function.

On Right Part

- The network extracts features and expands spatial support of the lower resolution feature maps to gather and assemble the necessary information to output a two-channel volumetric segmentation.



- At each stage, a **deconvolution** operation is employed to **increase the size of the inputs followed by one to three convolutional layers**, involving **half the number of $5 \times 5 \times 5$ kernels** applied in the previous layer.
- **The residual function** is learned, similar to left part of the network.
- The 2 features maps computed by **a very last convolutional layer**, having **$1 \times 1 \times 1$ kernel size** and producing **outputs of the same size as input volume**.
- These two output feature maps are **the probabilistic segmentation of the foreground and background regions by applying soft-max voxelwise**.

Q2. Highway Networks- Gating Function to highway

Answer:

It is found that difficulties are optimizing a very deep neural network. However, it's still an open problem with why it is difficult to optimize a deep network. (it is due to gradient vanishing problem.) Inspired by LSTM (Long Short-Term Memory), authors thereby **make use of gating function to adaptively bypass or transform the signal so that the network can go deeper**. The deep network with more than 1000 layers can also be optimized.

Plain Network

Before going into Highway Networks, Let us start with plain network which consists of L layers where the l -th layer (with omitting the symbol for the layer):

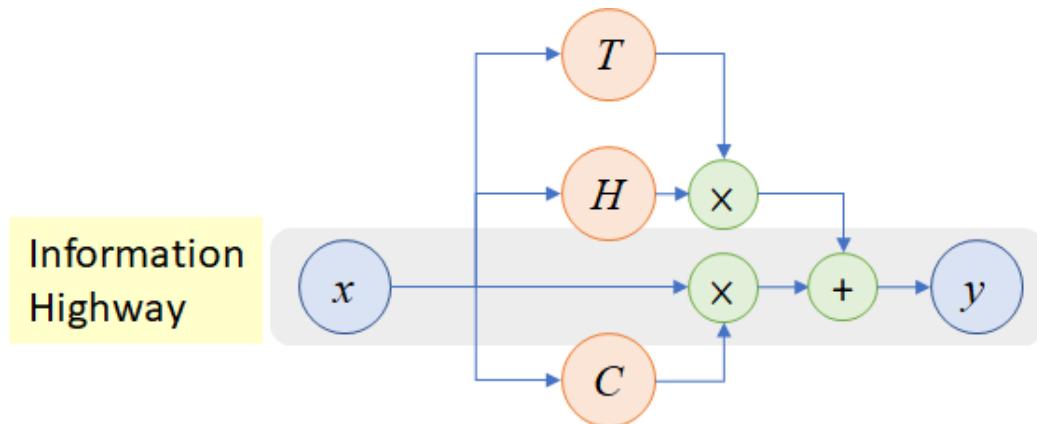
$$\mathbf{y} = H(\mathbf{x}, \mathbf{W}_H).$$

Where x is input, WH is the weight, H is the transform function followed by an activation function, and y is the output. And for i -th unit:

$$y_i = H_i(\mathbf{x})$$

We compute the y_i and pass it to the next layer.

Highway Network



In a highway network, 2 non-linear transforms T and C are introduced:

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W}_H) \cdot T(\mathbf{x}, \mathbf{W}_T) + \mathbf{x} \cdot C(\mathbf{x}, \mathbf{W}_C).$$

where **T is Transform Gate**, and **C is the Carry Gate**.

In particular, $C = 1 - T$:

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W}_H) \cdot T(\mathbf{x}, \mathbf{W}_T) + \mathbf{x} \cdot (1 - T(\mathbf{x}, \mathbf{W}_T)).$$

We can have below conditions for specific T values:

$$\mathbf{y} = \begin{cases} \mathbf{x}, & \text{if } T(\mathbf{x}, \mathbf{W}_T) = 0, \\ H(\mathbf{x}, \mathbf{W}_H), & \text{if } T(\mathbf{x}, \mathbf{W}_T) = 1. \end{cases}$$

When $T=0$, we pass input as output directly, which creates an information highway. That's why it is called the Highway Network.

When $T=1$, we use non-linear activated transformed input as output.

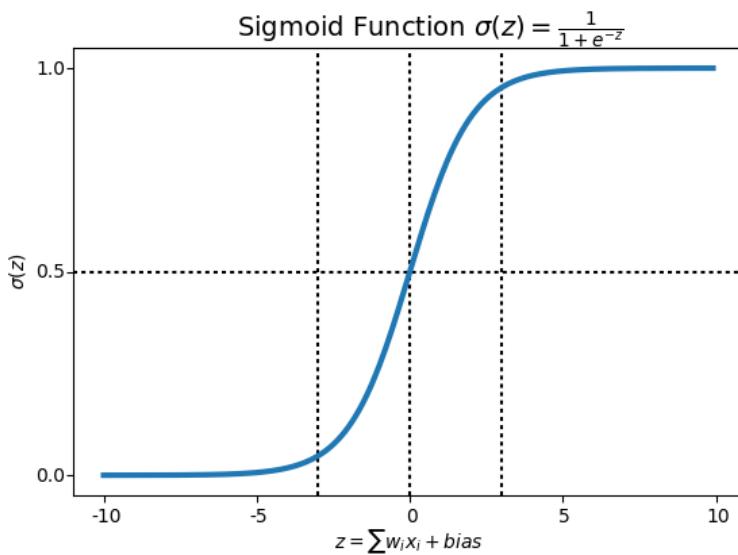
Here, in contrast to the i -th unit in plain network, the authors introduce the **block** concept. For i -th **block**, there is a **block state** $H_i(\mathbf{x})$, and **transform gate output** $T_i(\mathbf{x})$. And the corresponding **block output** y_i :

$$y_i = H_i(\mathbf{x}) * T_i(\mathbf{x}) + x_i * (1 - T_i(\mathbf{x}))$$

which is connected to the next layer.

- Formally, **$T(x)$ is the sigmoid function**:

$$f_{C+1}(z) = [(f_C \circ f_C)(z)] \oplus [\text{conv}(z)]$$



Sigmoid function caps the output between 0 to 1. When the input has a too-small value, it becomes 0. When the input has a too-large amount, it becomes 1. **Therefore, by learning WT and bT , a network can adaptively pass $H(x)$ or pass x to the next layer.**

And the author claims that this helps to have the simple initialization scheme for WT which is independent of nature of H .

bT can be initialized with the negative value (e.g., -1, -3, etc.) such that the network is initially biased towards carrying behaviour.

LSTM inspires the above idea as the authors mentioned.

And SGD(**Stochastic Gradient Descent**) did not stall for networks with more than 1000 layers. However, the exact results have not been provided.

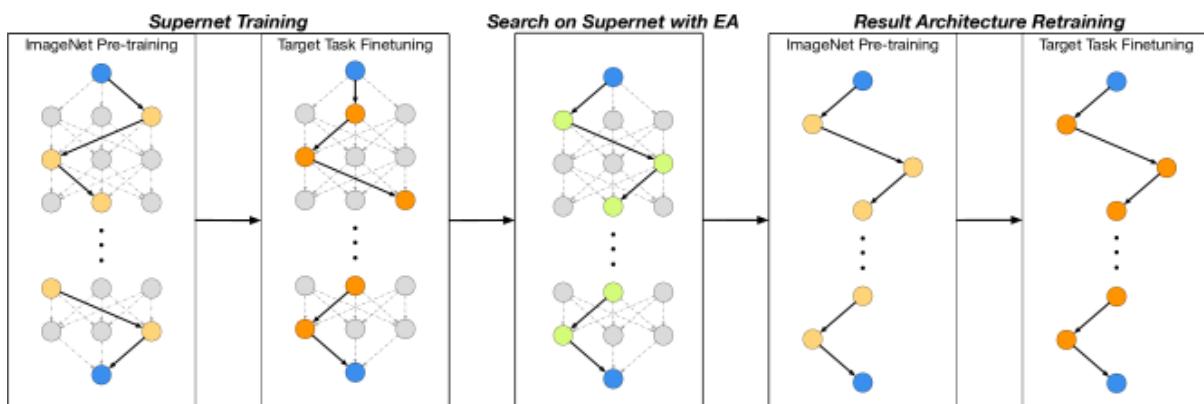
Q3. What is DetNAS: Neural Architecture Search(NAS) on Object Detection?

Answer:

Object detection is one of the most fundamental computer vision(OpenCV) tasks and has been widely used in real-world applications. The performance of object detectors highly relies on features extracted by backbones. However, most works on object detection directly use networks designed for classification as a backbone the feature extractors, e.g., ResNet. The architectures optimized on image classification can not guarantee performance on object detection. It is known that there is an essential gap between these two different tasks. Image classification basically focuses on "What" main object of the image is, while object detection aims at finding "Where" and "What" each object

instance in an image. There have been little works focusing on backbone design for object detector, except the hand-craft network, DetNet.

Neural architecture search (NAS) has achieved significant progress in image classification and semantic segmentation. The networks produced by search have reached or even surpassed the performance of the hand-crafted ones on this task. But object detection has never been supported by NAS before. Some NAS(Neural architecture search) work directly applies architecture searched on CIFAR-10 classification on object detection.



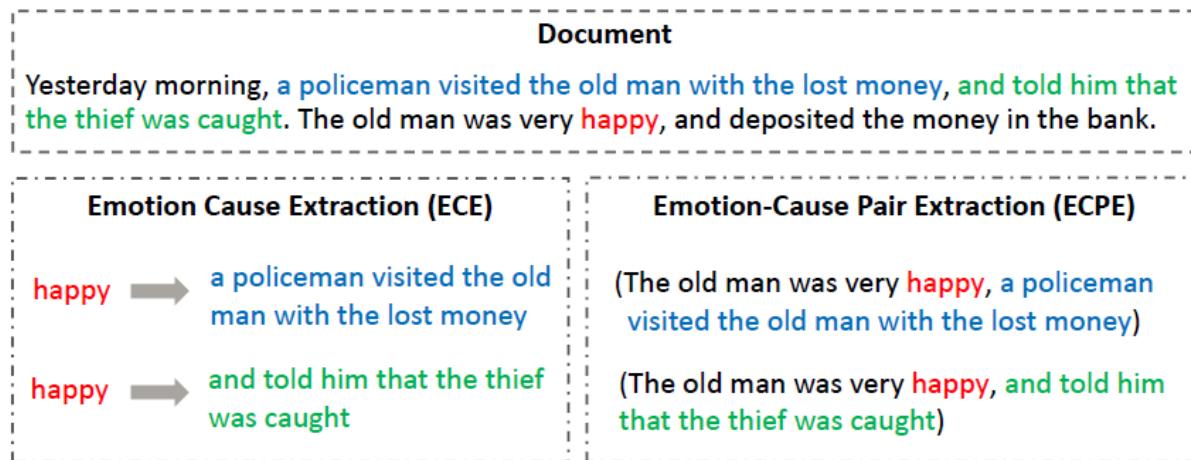
In this work, we present the first effort towards learning a backbone network for object detection tasks. Unlike previous NAS works, our method does not involve any architecture-level transfer. We propose DetNAS to conduct neural architecture search directly on the target tasks. The quests are even performed with precisely the same settings to the target task. Training an object detector usually needs several days and GPUs, no matter using a pre-train-finetune scheme or training from scratch. Thus, it is not affordable to directly use reinforcement learning (RL) or evolution algorithm (EA) to search the architectures independently. To overcome this obstacle, we formulate this problem into searching the optimal path in the large graph or supernet. In simple terms, DetNAS consists of three steps: (1) training a supernet that includes all sub-networks in search space; (2) searching for the sub-network with the highest performance on the validation set with EA; (3) retraining the resulting network and evaluating it on the test set.

Q4. You have any idea about ECE (Emotion cause extraction).

Answer:

Emotion cause extraction (ECE) aims at extracting potential causes that lead to emotion expressions in the text. The ECE task was first proposed and defined as a word-level sequence labeling problem in Lee et al. To solve the shortcoming of extracting causes at the word level, Gui et al. 2016 released a new corpus which has received much attention in the following study and becomes a benchmark dataset for ECE research.

Below Fig. Displays an example from this corpus, there are five clauses in a document. The emotion “happy” is contained in fourth clause. We denote this clause as an *emotion clause*, which refers to a term that includes emotions. It has two corresponding causes: “a policeman visited the old man with the lost money” in the second clause and, “told him that the thief was caught” in the third clause. We name them as *cause clause*, which refers to a term that contains causes.



In this work, we propose a new task: emotion-cause pair extraction (ECPE), which aims to extract all potential pairs of emotions and corresponding causes in the document. In Above Fig, we show the difference between the traditional ECE task and our new ECPE task. The goal of ECE is to extract the corresponding cause clause of the given emotion. In addition to a document as the input, ECE needs to provide annotated feeling at first before cause extraction.

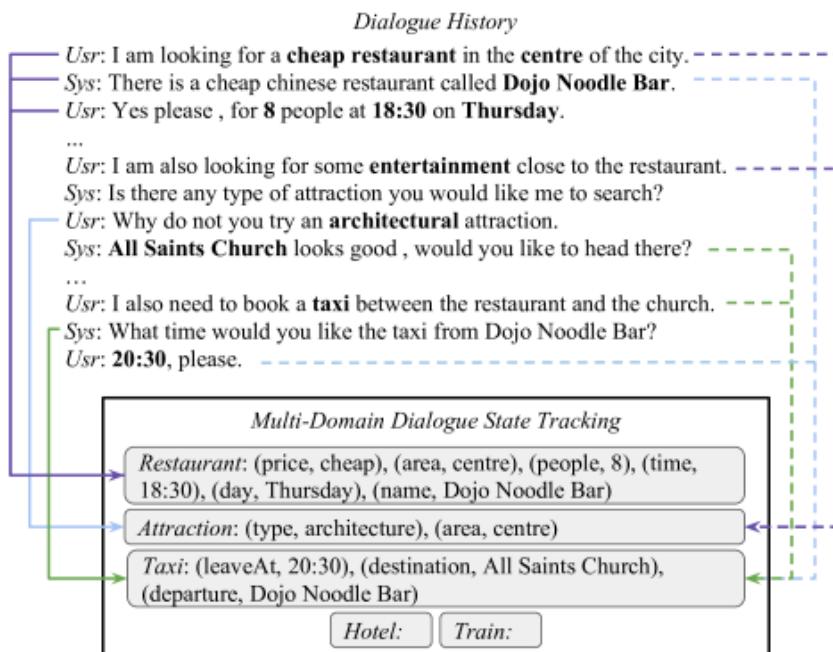
In contrast, the output of our ECPE task is a pair of emotion-cause, without the need of providing emotion annotation in advance. From Above fig., e.g., given the annotation of feeling: “happy,” the goal of ECE is to track the two corresponding cause clauses: “a policeman visited the old man with the lost money” and “and told him that the thief was caught.” While in the ECPE task, the goal is to directly extract all pairs of emotion clause and cause clause, including (“The old man was delighted”, “a policeman visited the old man with the lost money”) and (“The old man was pleased”, “and told him that the thief was caught”), without providing the emotion annotation “happy”.

To address this new ECPE task, we propose a two-step framework. Step 1 converts the emotion-cause pair extraction task to two individual sub-tasks (emotion extraction and cause extraction respectively) via two kinds of multi-task learning networks, intending to extract a set of emotion clauses and a set of cause clauses. Step 2 performs emotion-cause pairing and filtering. We combine all the elements of the two sets into pairs and finally train a filter to eliminate the couples that do not contain a causal relationship.

Q5.What is DST (Dialogue state tracking)?

Answer:

Dialogue state tracking (DST) is a core component in task-oriented dialogue systems, such as restaurant reservations or ticket bookings. The goal of DST is to extract user goals expressed during conversation and to encode them as a compact set of the dialogue states, i.e., a set of slots and their corresponding values. E.g., as shown in below fig., *(slot, value)* pairs such as *(price, cheap)* and *(area, centre)* are extracted from the conversation. Accurate DST performance is important for appropriate dialogue management, where user intention determines the next system action and the content to query from the databases.



State tracking approaches are based on the assumption that ontology is defined in advance, where all slots and their values are known. Having a predefined ontology can simplify DST into a classification problem and improve performance (Henderson et al., 2014b; Mrkšić et al., 2017; Zhong et al., 2018). However, there are two significant drawbacks to this approach: 1) A full ontology is hard to obtain in advance (Xu and Hu, 2018). In the industry, databases are usually exposed through an external API only, which is owned and maintained by others. It is not feasible to gain access to enumerate all the possible values for each slot. 2) Even if a full ontology exists, the number of possible slot values could be significant and variable. For example, a restaurant name or a train departure time can contain a large number of possible values. Therefore, many of the previous works that are based on neural classification models may not be applicable in real scenarios.

Q6.What is NMT(Neural machine translation)?

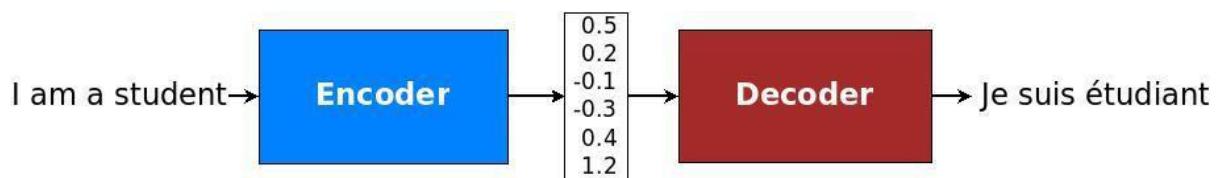
Answer:

NMT stands for Neural machine translation, which is the use of neural network models to learn the statistical model for machine translation.

The key benefit to the approach is that the single system can be trained directly on the source and target text, no longer requiring the pipeline of specialized methods used in statistical (ML) machine learning.

Unlike the traditional phrase-based translation system which consists of many sub-components that are tuned separately, neural machine translation attempts to build and train a single, large neural network that reads a sentence and outputs a correct translation.

As such, neural machine translation(NMT) systems are said to be end-to-end systems as only one model is required for the translation.



In Encoder

The task of the encoder is to provide the representation of a input sentence. The input sentence is a sequence of words, for which we first consult embedding matrix. Then, as in the primary language model described previously, we process these words with a recurrent neural network(RNN). This results in hidden states that encode each word with its left context, i.e., all the preceding words. To also get the right context, we also build a recurrent neural network(RNN) that runs right-to-left, or, from the end of the sentence to beginning. Having two recurrent neural networks(RNN) running in two directions is known as the bidirectional recurrent neural network(RNN).

In Decoder

The decoder is the recurrent neural network(RNN). It takes some representation of input context (more on that in the next section on the attention mechanism) and previous hidden state and the output word prediction, and generates a new hidden decoder state and the new output word prediction.

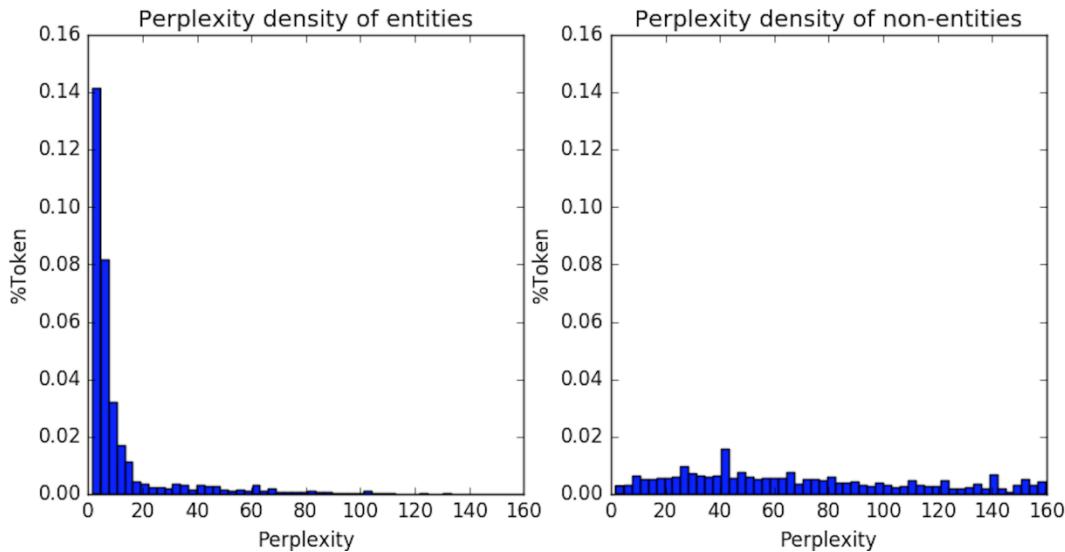
If you use LSTMs for the encoder, then you also use LSTMs for the decoder. From hidden state. You now predict the output word. This prediction takes the form of the probability distribution over entire output vocabulary. If you have a vocabulary of, say, 50,000 words, then the prediction is a 50,000 dimensional vector, each element corresponding to the probability predicted for one word in the vocabulary.

Q7. What is Character-Level models (CLM)?

Answer:

In English, there is strong empirical evidence that the character sequence that create up proper nouns tend to be distinctive. Even divorced of context, human reader can predict that “hoekstenberger” is an entity, but “abstractually” is not. Some NER research explores use of character-level features

including capitalization, prefixes and suffixes Cucerzan and Yarowsky; Ratinov and Roth (2009), and character-level models (CLMs) Klein et al. (2003) to improve the performance of NER, but to date there has been no systematic study isolating utility of CLMs in capturing the distinctions between name and non-name tokens in English or across other languages.



We conduct the experimental assessment of the discriminative power of CLMs for a range of languages: English, Arabic, Amharic, Bengali, Farsi, Hindi, Somali, and Tagalog. These languages use the variety of scripts and orthographic conventions (e.g, only three use capitalization), come from different language families, and vary in their morphological complexity. We represent the effectiveness of CLMs(character-level models) in distinguishing name tokens from non-name tokens, as illustrated by the above Figure, which shows confusion in histograms from a CLM trained on entity tokens. Our models use individual tokens, but perform extremely well in spite of taking no account of the word context.

We then assess the utility of directly adding simple features based on this CLM(character-level model) implementation to an existing NER system, and show that they have the significant positive impact on performance across many of the languages we tried. By adding very simple CLM-based features to the system, our scores approach those of a state-of-the-art(SOTA) NER system Lample et al. (2016) across multiple languages, representing both the unique importance and broad utility of this approach.

Q8.What is LexNLP package?

Answer:

Over the last 2 decades, many high-quality, open-source packages for natural language processing(NLP) and machine learning(ML) have been released. Developers and researchers can quickly write applications in languages such as Python, Java, and R that stand on shoulders of

comprehensive, well-tested libraries such as Stanford NLP (Manning et al. (2014)), OpenNLP (ApacheOpenNLP (2018)), NLTK (Bird et al. (2009)), spaCy (Honnibal and Montani (2017), scikit-learn library (Buitinck et al. (2013), Pedregosa et al. (2011)), and Gensim (Řehůřek and Sojka (2010)). Consequently, for most of the domains, rate of research has increased and cost of the application development has decreased.

For some specialized areas like marketing and medicines, there are focused libraries and organizations like BioMedICUS (Consortium (2018)), RadLex (Langlotz (2006)), and the Open Health Natural Language Processing(NLP) Consortium. Law, however, has received substantially less attention than others, despite its ubiquity, societal importance, and the specialized form. LexNLP is designed to fill this gap by providing both tools and data for developers and researchers to work with real legal and regulatory text, including statutes, regulations, the court opinions, briefs, contracts, and the other legal work products.

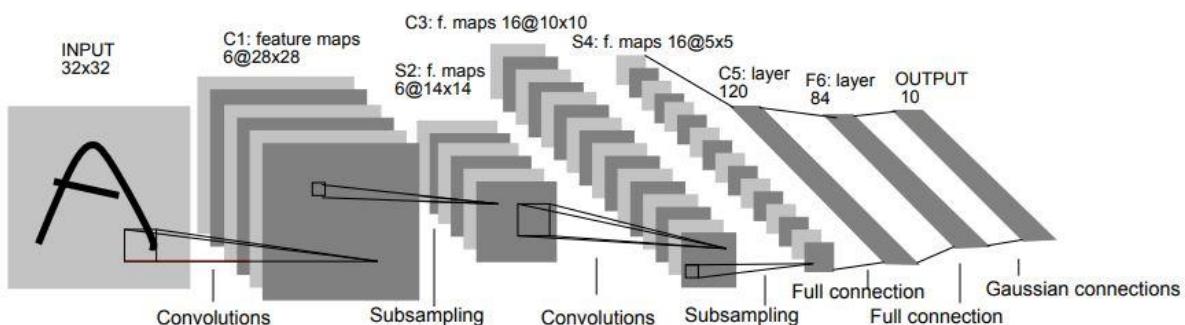
Law is the domain driven by language, logic, and the conceptual relationships, ripe for computation and analysis (Ruhl et al. (2017)). However, in our experience, natural language processing(NLP) and machine learning(ML) have not been applied as fruitfully or widely in legal as one might hope. We believe that the key impediment to academic and commercial application has been lack of tools that allow users to turn the real, unstructured legal document into structured data objects. The Goal of LexNLP is to make this task simple, whether for the analysis of statutes, regulations, court opinions, briefs or the migration of legacy contracts to smart contract or distributed ledger systems.

Q9.Explain The Architecture of LeNet-5.

Answer:

Yann LeCun, Leon Bottou, Yoshua Bengio and Patrick Haffner proposed the neural network architecture for the handwritten and machine-printed character recognition in the 1990's which they called them LeNet-5. The architecture is straightforward and too simple to understand that's why it is mostly used as a first step for teaching (CNN)Convolutional Neural Network.

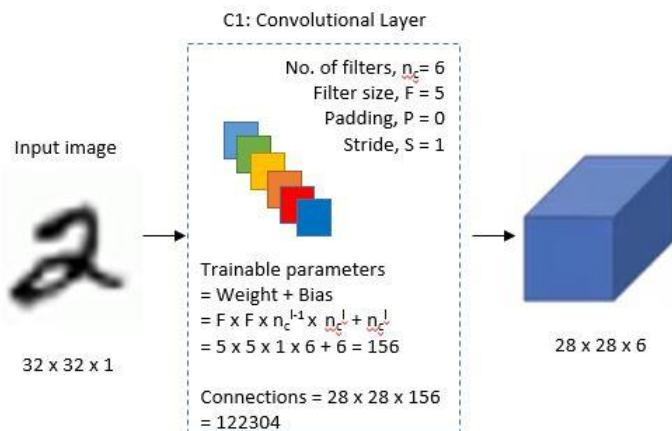
Architecture



This architecture consists of two sets of convolutional and average pooling layers, followed by the flattening convolutional layer, then 2 fully-connected layers and finally the softmax classifier.

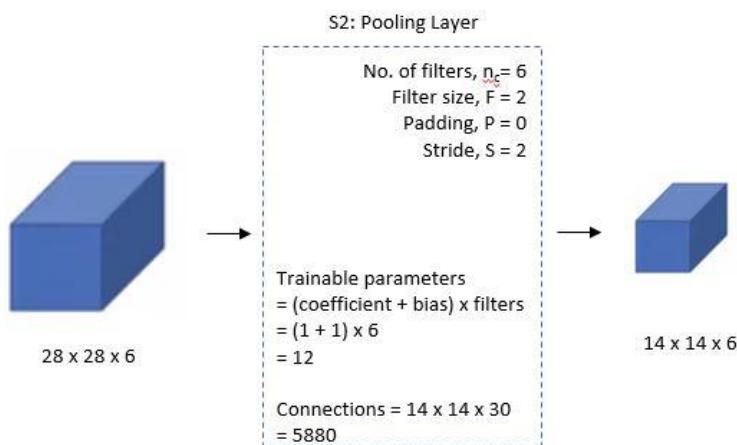
In the First Layer:

The input for LeNet-5 is the 32×32 grayscale image which passes through first convolutional layer with 6 feature maps or filters having size 5×5 and the stride of one. Image dimensions changes from $32 \times 32 \times 1$ to $28 \times 28 \times 6$.



In Second Layer:

Then it applies average pooling layer or sub-sampling layer with the filter size 2×2 and stride of two. The resulting image dimension will be reduced to $14 \times 14 \times 6$.



Third Layer:

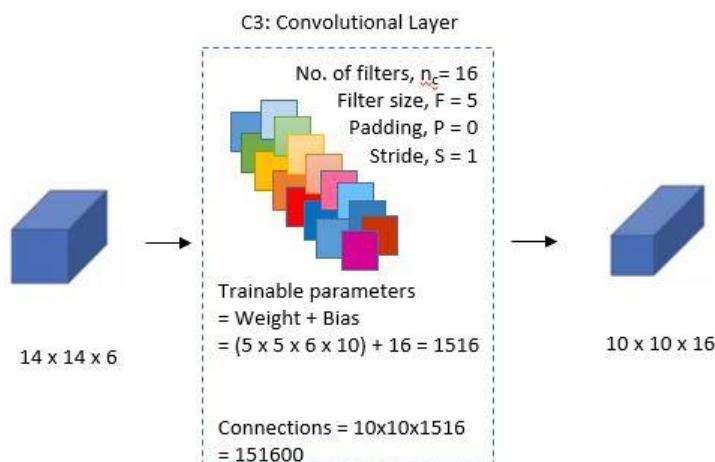
Next, there is the second convolutional layer with 16 feature maps having size 5×5 and the stride of 1. In this layer, only ten out of sixteen feature maps are connected to 6 feature maps of previous layer as shown below.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X			X	X	X			X	X	X	X		X	X	
1	X	X			X	X	X			X	X	X	X		X	
2	X	X	X			X	X	X		X		X	X	X		
3		X	X	X		X	X	X	X		X		X	X	X	
4			X	X	X		X	X	X	X		X	X		X	
5				X	X	X		X	X	X	X		X	X	X	

TABLE I

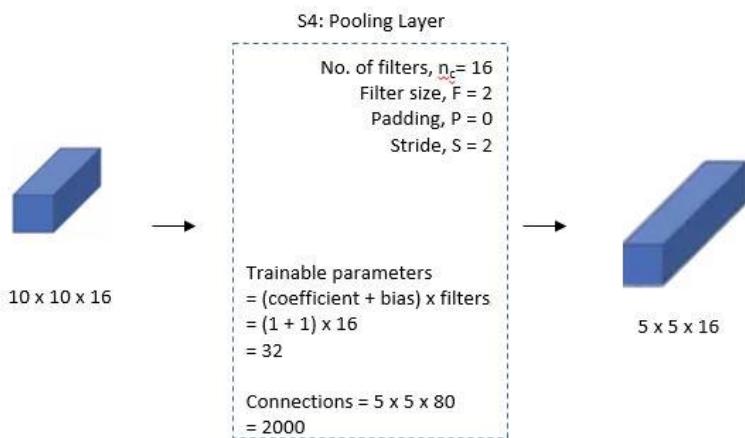
EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

The main reason is to break symmetry in the network and keeps a number of connections within reasonable bounds. That is why the number of training parameters in this layers are 1516 instead of 2400 and similarly, number of connections are 151600 instead of 240000.



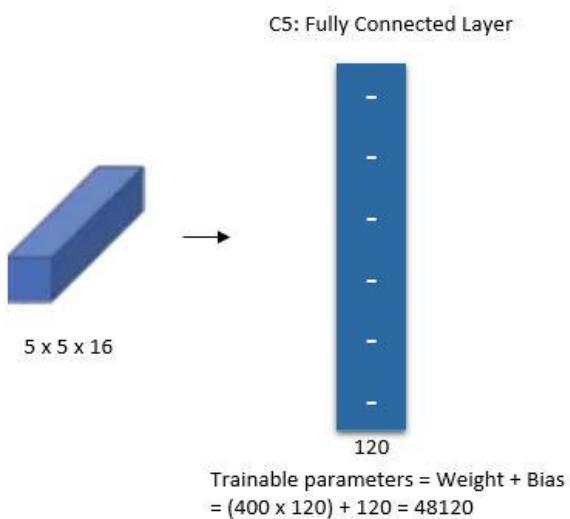
Fourth Layer:

In the fourth layer (S4) is an average pooling layer with filter size 2×2 and stride of 2. This layer is same as second layer (S2) except it has 16 feature maps so output will be reduced to $5 \times 5 \times 16$.



Fifth Layer:

The fifth layer (C5) is the fully connected convolutional layer with 120 feature maps each of the size 1x1. Each of 120 units in C5 is connected to all the 400 nodes ($5 \times 5 \times 16$) in the fourth layer S4.



Sixth Layer:

The sixth layer is also fully connected layer (F6) with 84 units.

C5: Fully Connected Layer



F6: Fully Connected Layer



$$\begin{aligned}\text{Trainable parameters} &= \text{Weight} + \text{Bias} \\ &= (400 \times 120) + 120 = 48120\end{aligned}$$

$$\begin{aligned}\text{Trainable parameters} &= \text{Weight} + \text{Bias} \\ &= (120 \times 84) + 84 = 10164\end{aligned}$$

Output Layer:

Finally, there is fully connected softmax output layer \hat{y} with 10 possible values corresponding to digits from 0 to 9.

F6: Fully Connected Layer



Output

- 0
- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9



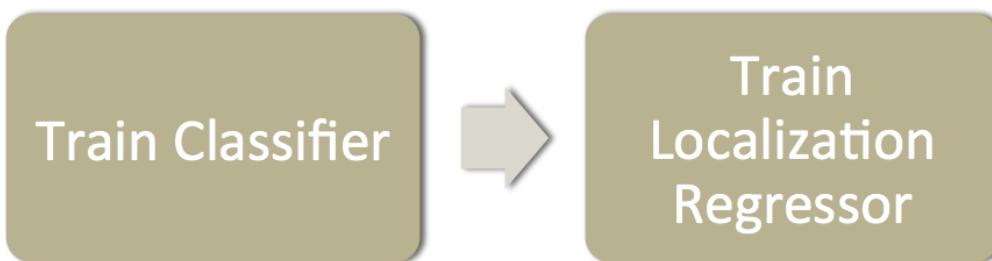
$$\begin{aligned}\text{Trainable parameters} &= \text{Weight} + \text{Bias} \\ &= (120 \times 84) + 84 = 10164\end{aligned}$$

Q1.Explain Overfeat in Object detection.

Answer:

Overfeat: It is a typical model of integrating object detection, localization, and classification tasks whole into one convolutional neural network(CNN). The main idea is to do image classification at different locations on regions of multiple scales of the image in a sliding window fashion, and second, predict bounding box locations with the regressor trained on top of the same convolution layers.

This model architecture is too similar to AlexNet. This model is trained as follows:



1. Train a CNN model (identical to AlexNet) on image classification tasks.
2. Then, we replace top classifier layers by the regression network and trained it to predict object bounding boxes at each spatial location and scale. Regressor is class-specific, each generated for one class image.
 - Input: Images with classification and bounding box.
 - Output: $(x_{left}, x_{right}, y_{top}, y_{bottom})$, 4 values in total, representing the coordinates of the bounding box edges.
 - Loss: The regressor is trained to minimize L_2 norm between the generated bounding box and truth for each training example.

At the detection time,

1. It Performs classification at each location using the pretrained CNN model.
2. It Predicts object bounding boxes on all classified regions generated by the classifier.
3. Merge bounding boxes with sufficient overlap from localization and sufficient confidence of being the same object from the classifier.

Q2. What is Multipath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction?

Answer:

In this paper, we focus on problem of predicting future agent states, which is the crucial task for robot planning in real-world environments. We are specifically interested in addressing this problem for self-driving vehicles, application with a potentially enormous societal impact. Mainly, predicting the future of other agents in this domain is vital for safe, comfortable, and efficient operation. E.g., it is important to know whether to yield to the vehicle if they are going to cut in front of our robot or when would be the best time to add into traffic. Such future prediction requires an understanding of a static and dynamic world context: road semantics (*like* lane connectivity, stop lines), traffic light informations, and past observations of other agents, as in below Fig.

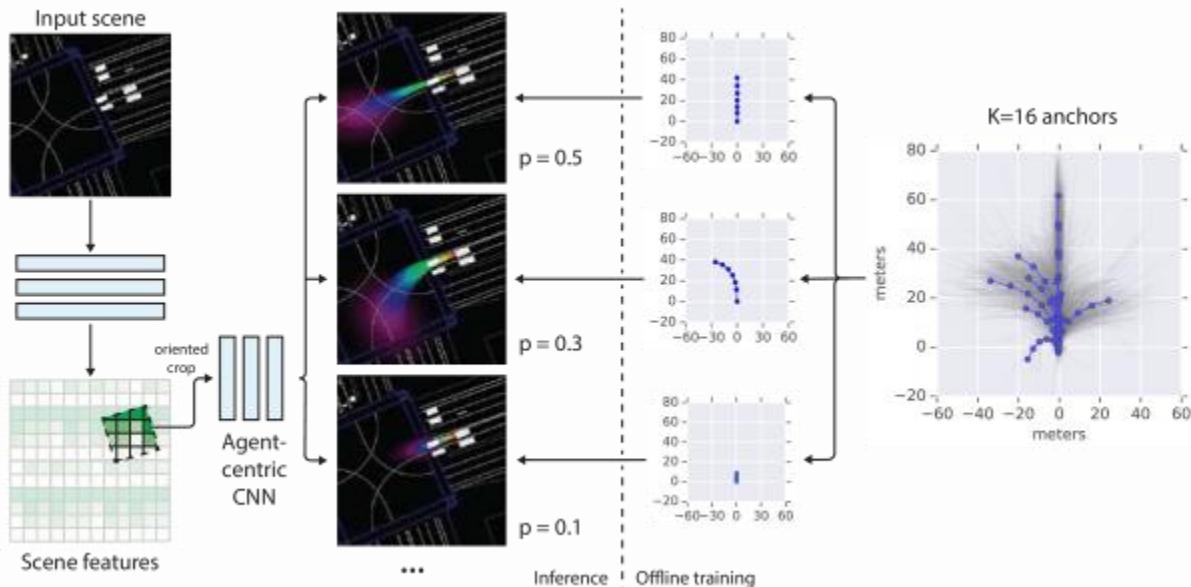
A fundamental aspect of the future state prediction is that it is inherently *stochastic*, as agents can't know each other's motivations. When we are driving, we can never really be sure what other drivers will do next, and it is essential to consider multiple outcomes and their likelihood.

We seek the model of the future that can provide both (i) a weighted, parsimonious set of discrete trajectories that covers space of likely outcomes and (ii) a closed-form evaluation of the likelihood of any trajectory. These two attributes enable efficient reasoning in relevant planning use-cases, e.g., human-like reactions to discrete trajectory hypotheses (*e.g.*, yielding, following), and probabilistic queries such as the expected risk of collision in a space-time region.

This model addresses these issues with critical insight: it employs a fixed set of *trajectory anchors* as the basis of our modeling. This lets us factor stochastic uncertainty hierarchically: First, *intent uncertainty* captures the uncertainty of *what* an agent intends to do and is encoded as a distribution over the set of anchor trajectories. Second, given an intent, *control uncertainty* represents our uncertainty over *how* they might achieve it. We assume control uncertainty is normally distributed at each future time step [Thrun05], parameterized such that the mean corresponds to a context-specific offset from the anchor state, with the associated covariance capturing the unimodal aleatoric uncertainty [Kendall17]. In Fig. Illustrates a typical scenario where there are three likely intents given the scene context, with control mean offset refinements respecting road geometry, and control uncertainty intuitively growing over time.

Our trajectory anchors are modes found in our training data in state-sequence space via unsupervised learning. These anchors provide templates for coarse-granularity futures for an agent and might correspond to semantic concepts like “change lanes,” or “slow down” (although to be clear, we don’t use any semantic concepts in our modeling).

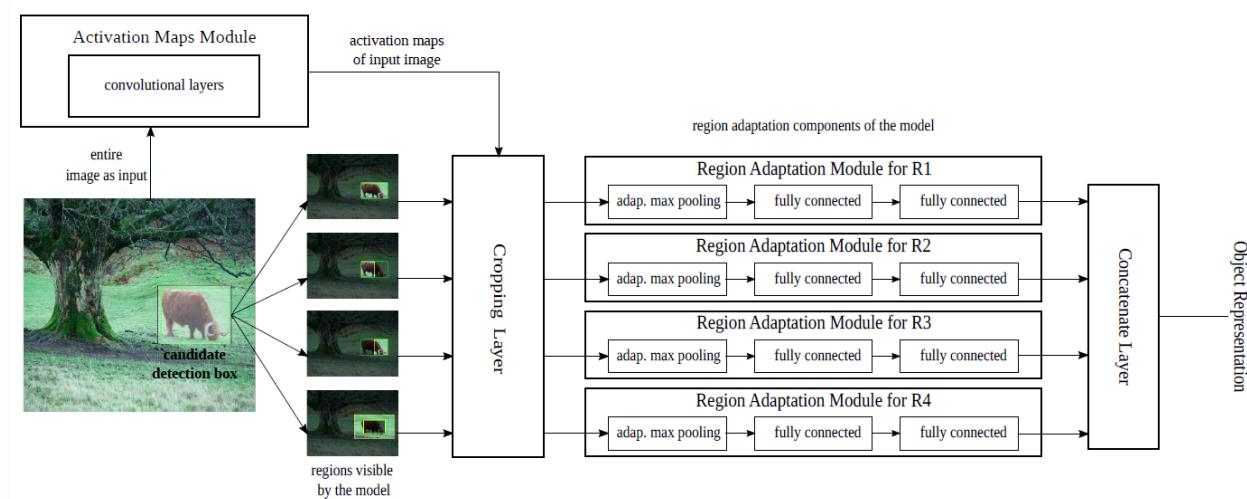
Our complete model predicts a Gaussian mixture model (GMM) at each time step, with the mixture weights (intent distribution) fixed over time. Given such a parametric distribution model, we can directly evaluate the likelihood of any future trajectory and have a simple way to obtain a compact, diverse weighted set of trajectory samples: the MAP sample from each anchor-intent.



Q3. An Object detection approach using MR-CNN

Answer:

Multi-Region CNN (MR-CNN): Object representation using multiple regions to capture several different aspects of one object.

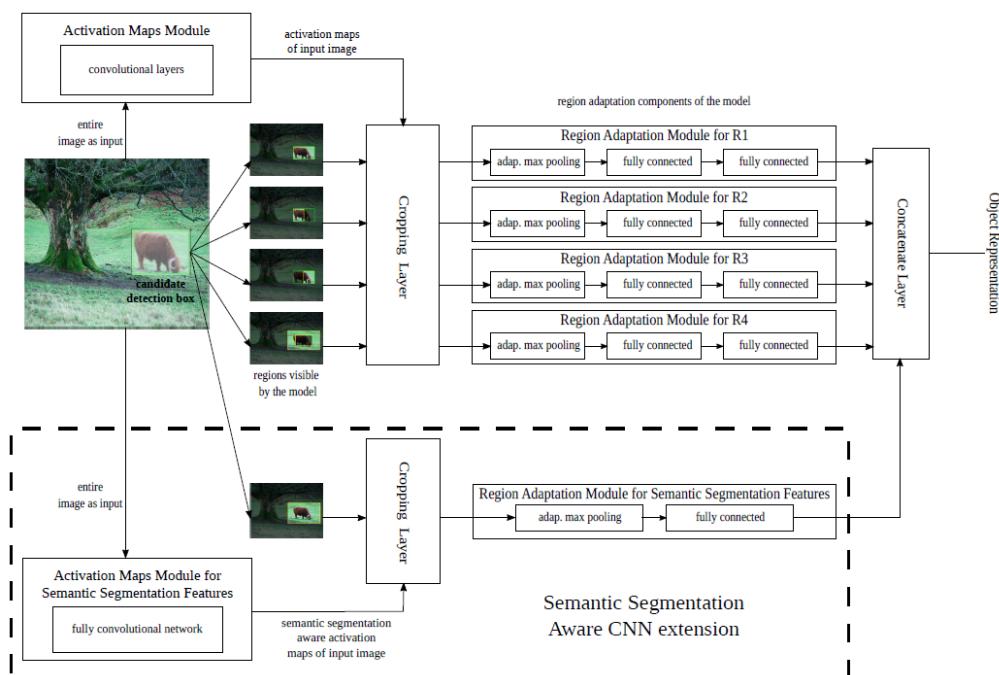


Network Architecture of MR-CNN

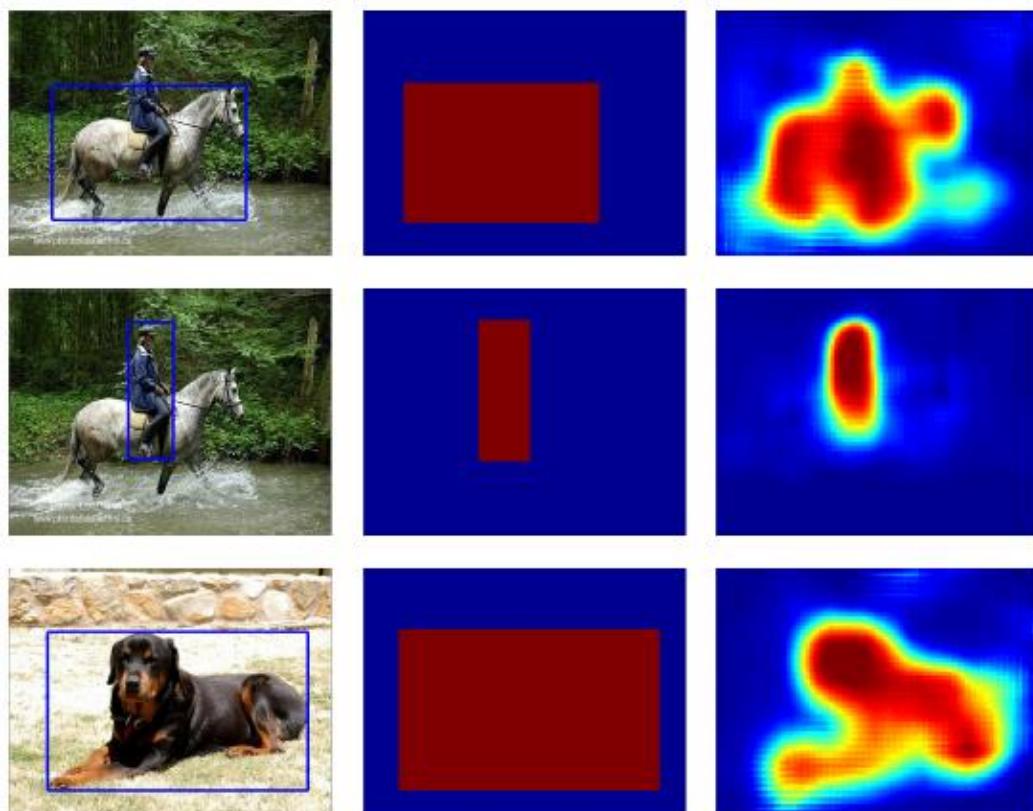
- First, the input image goes through Activation Maps Module, as shown above, and outputs the activation map.
- Bounding box** or Region proposals candidates are generated using Selective Search.
- For each bounding box candidate B , a set of regions $\{R_i\}$, with $i=1$ to k , are generated, that is why it is known as multi-region. More details about the choices of multiple areas are described in next sub-section.
- ROI pooling is performed for each region R_i , cropped or pooled area goes through fully connected (FC) layers, at each Region Adaptation Module.
- Finally, the output from all FC layers are added together to form a 1D feature vector, which is an object representation of the bounding box B .
- Here, VGG-16 ImageNet pre-trained model is used. The max-pooling layer after the last conv layer is removed.

Q4. Object detection using Segmentation-aware CNN

Answer:



- There are close connections between segmentation and detection. And segmentation related ques are empirically known to help object detection often.
- Two modules are added: **Activation maps module for semantic segmentation-aware features**, and **regions adaptation module for grammarly segmentation-aware feature**.
- There is no additional annotation used for training here.
- FCN is used for an activation map module.
- The last FC7 layer channels number is changed from 4096 to 512.



- The **weakly supervised training** strategy is used. **Artificial foreground class-specific segmentation mask is created using bounding box annotations.**

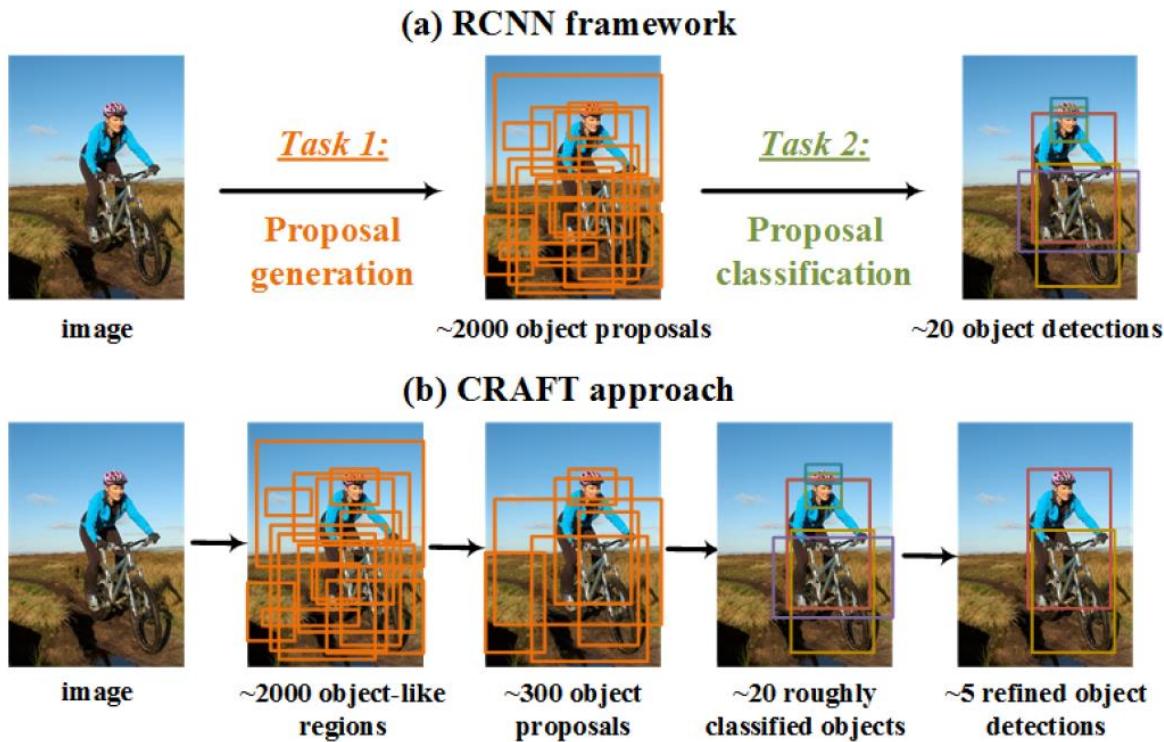
- More particularly, the ground truth bounding boxes of an image are projected on the spatial domain of the last hidden layer of the [FCN](#), and the "pixels" that lay inside the projected boxes are labelled as foreground while the rest are labelled as background.
- After training the FCN using the mask, the last classification layer is dropped. Only the rest of FCN is used.
- Though it is weakly supervised training, the foreground probabilities shown as above still carry some information, as shown above.
- The bounding box used is $1.5\times$ larger than the original bounding box.

Q5. What is CRAFT (Object detection)?

Answer:

CRAFT stands for Cascade Region-proposal-network **And FasT R-CNN**. It is reviewed by the Chinese Academy of Sciences **and** Tsinghua University. In Faster R-CNN, region proposal network is used to generate proposals. These proposals, after ROI pooling, are going through network for classification. However, CRAFT is found that there is a core problem in Faster R-CNN:

- In proposal generation, there is still a large proportion of background regions. The existence of many background sample causes many false positives.



In CRAFT(Cascade Region-proposal-network), as shown above, another CNN(Convolutional neural network) is added after RPN to generate fewer proposals (i.e., 300 here). Then, classification is performed on 300 proposals and outputs about 20 first detection results. For each primitive result, refined object detection is performed using one-vs-rest classification.

Cascade Proposal Generation

Baseline RPN

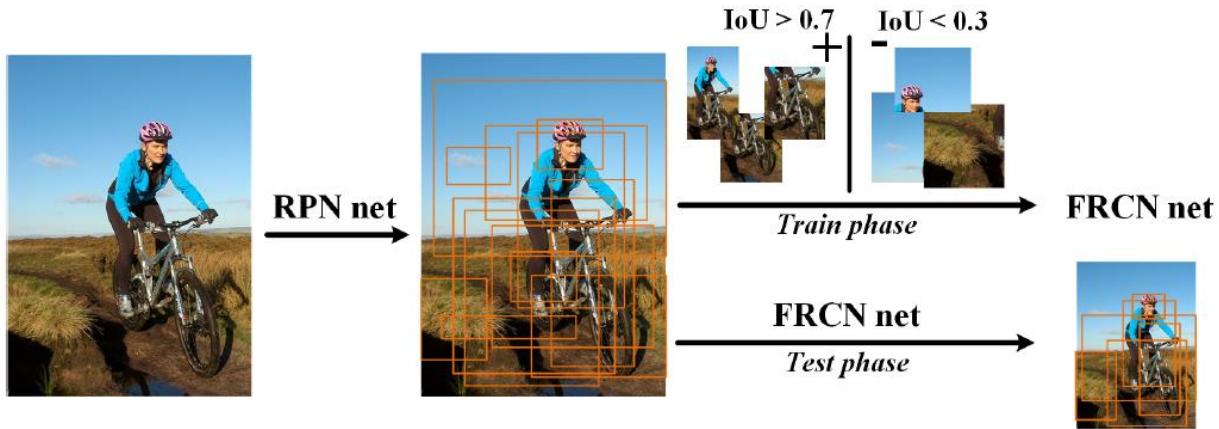
- An ideal proposal generator should generate as few proposal as possible while covering almost all object instances. Due to resolution loss caused by CNN pooling operation and the fixed aspect ratio of the sliding window, RPN is weak at covering objects with extreme shapes or scales.

aero	bike	bird	boat	bottle
95.44	98.81	93.90	92.78	80.38
bus	car	cat	chair	cow
98.12	96.00	99.16	91.80	99.18
table	dog	horse	mbike	persn
95.15	99.59	97.70	96.31	95.49
plant	sheep	sofa	train	tv
86.87	98.76	98.74	97.52	90.58

Recall Rates (is in %), Overall is 94.87%, lower than 94.87% is bold in the text.

- The above results are baseline RPN based on VGG_M trained using PASCAL VOC 2007 train+val, and tested on the test set.
- The recall rate on each object category varies a lot. Object with extreme aspect ratio and scale are hard to be detected, such as boat and bottle.

Proposed Cascade Structure



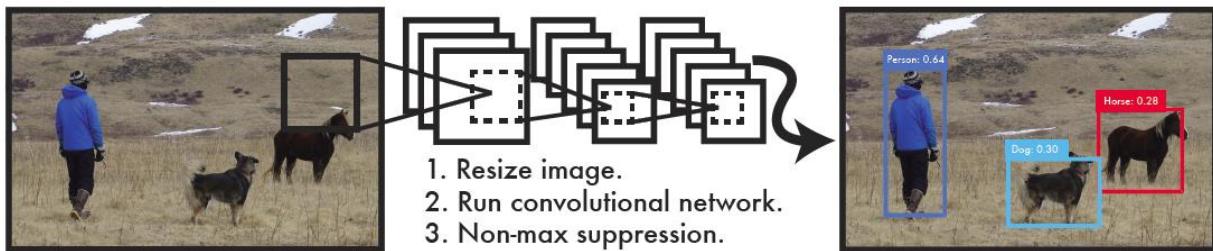
The concatenation classification network after RPN is denoted as FRCN Net here

- An additional classification network that comes after RPN.
- The additional network is the 2- class detection network denoted as FRCN net in above figure. It uses output of RPN as training data.
- After RPN net is trained, the 2000 first proposals of each training image are used as training data for the FRCN net.
- During training, +ve and -ve sampling are based on 0.7 IoU for negatives and below 0.3 IoU for negatives, respectively.
- **There are 2 advantages:**
 - 1) First, additional FRCN net further **improves quality of the object proposals** and **shrinks more background regions**, making proposals fit better with task requirement.
 - 2) Second, **proposals from multiple sources can be merged** as the input of the FRCN net so that complementary information can be used.

Q6. Explain YOLOv1 for Object Detection.

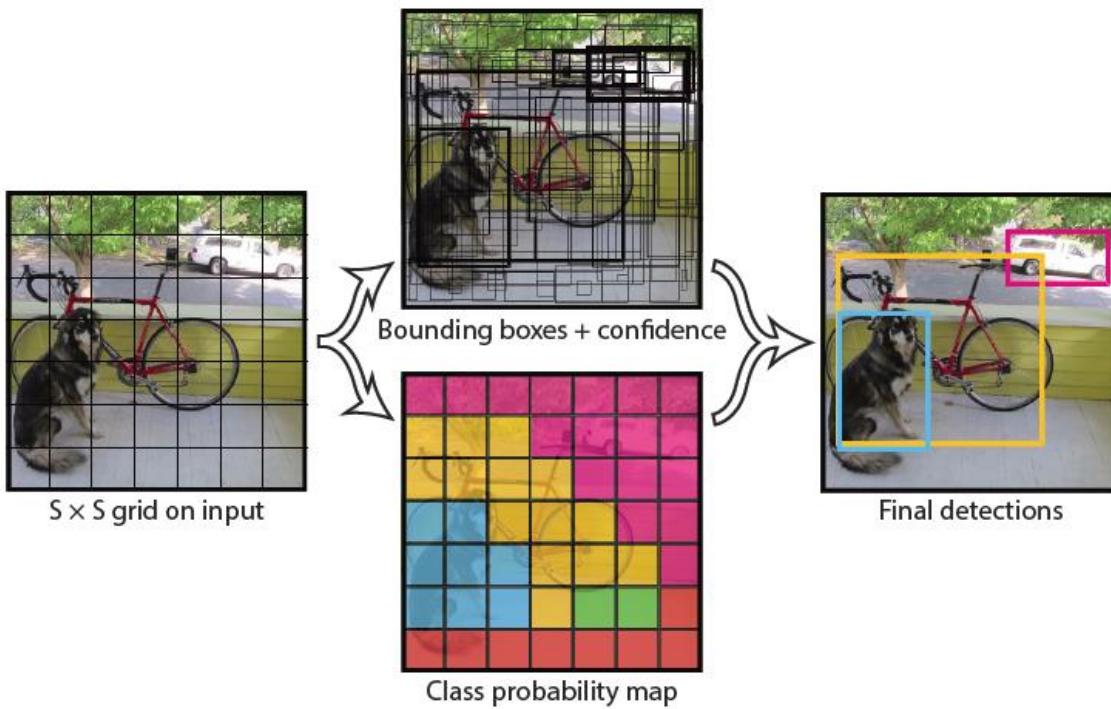
Answer:

YOLOv1 stands for You Look Only Once, it is reviewed by FAIR (Facebook AI Research). The network only looks at the image once to detect multiple objects.



By just looking image once, the detection speed is in real-time (45 fps). Fast YOLOv1 achieves 155 fps.

YOLO suggests having a unified network to perform all at once. Also, an end-to-end training network can be achieved.



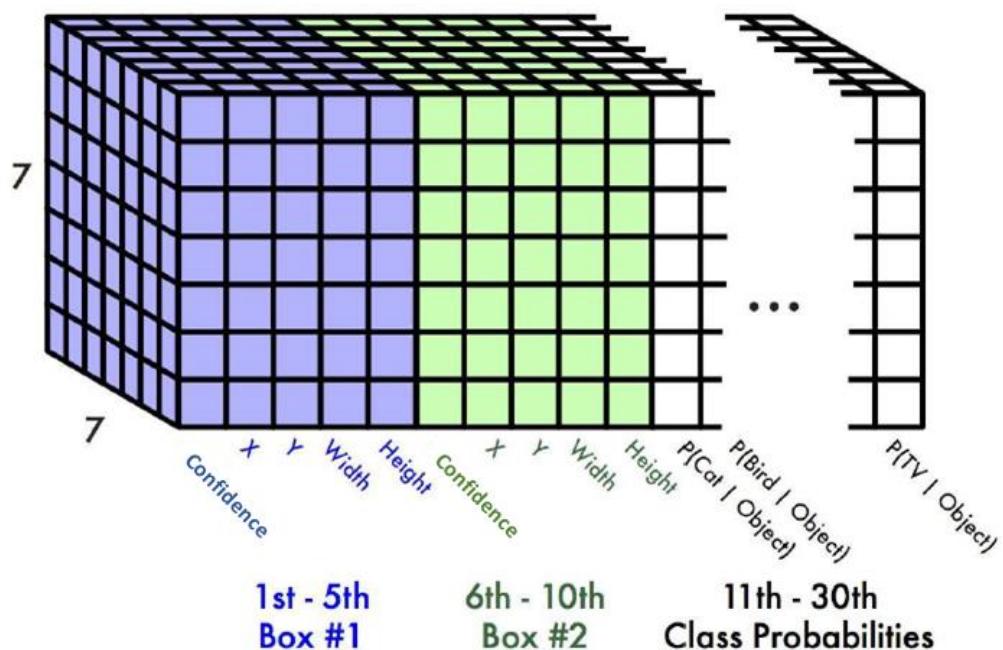
The input image is divided into the $S \times S$ grid ($S=7$). If the center of the object falls into the grid cell, that grid cell is responsible for detecting that object.

Each grid cell predict B bounding boxes ($B=2$) and confidence scores for those boxes. These confidence score reflect how confident model is that the box contains an object, i.e., any objects in the box, $P(\text{Objects})$.

Each bounding box consists of five predictions: x, y, w, h , and confidence.

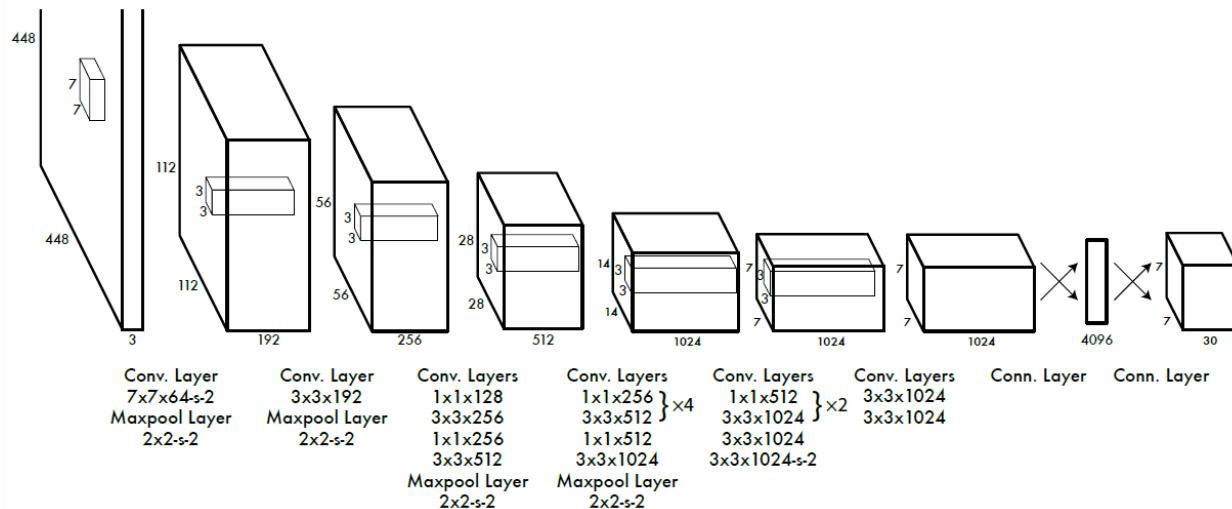
- The (x, y) coordinates represent center of the box relative to the bound of the grid cell.
- The height h and width w are predicted relative to whole image.
- The confidence represents the IOU (Intersection Over Union) between the predicted box and any ground truth box.

Each grid cell also predicts conditional class probabilities, $P(\text{Class}|\text{Object})$. (Total number of classes=20)



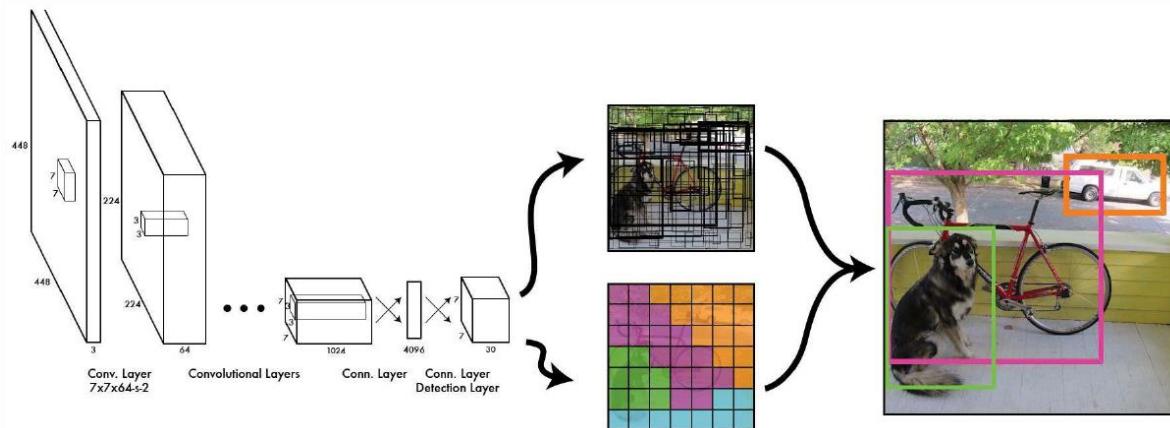
The output size becomes: $7 \times 7 \times (2 \times 5 + 20) = 1470$

Network Architecture of YOLOv1



The model consists of 24 convolutional layers, followed by two fully connected layers. Alternating 1×1 convolutional layers reduce features space from preceding layers. (1×1)Conv has been used in GoogLeNet for reducing the number of parameters.)

Fast YOLO fewer convolutional layers (9 instead of 24) and fewer filters in those layers. The network pipeline is summarized like below:

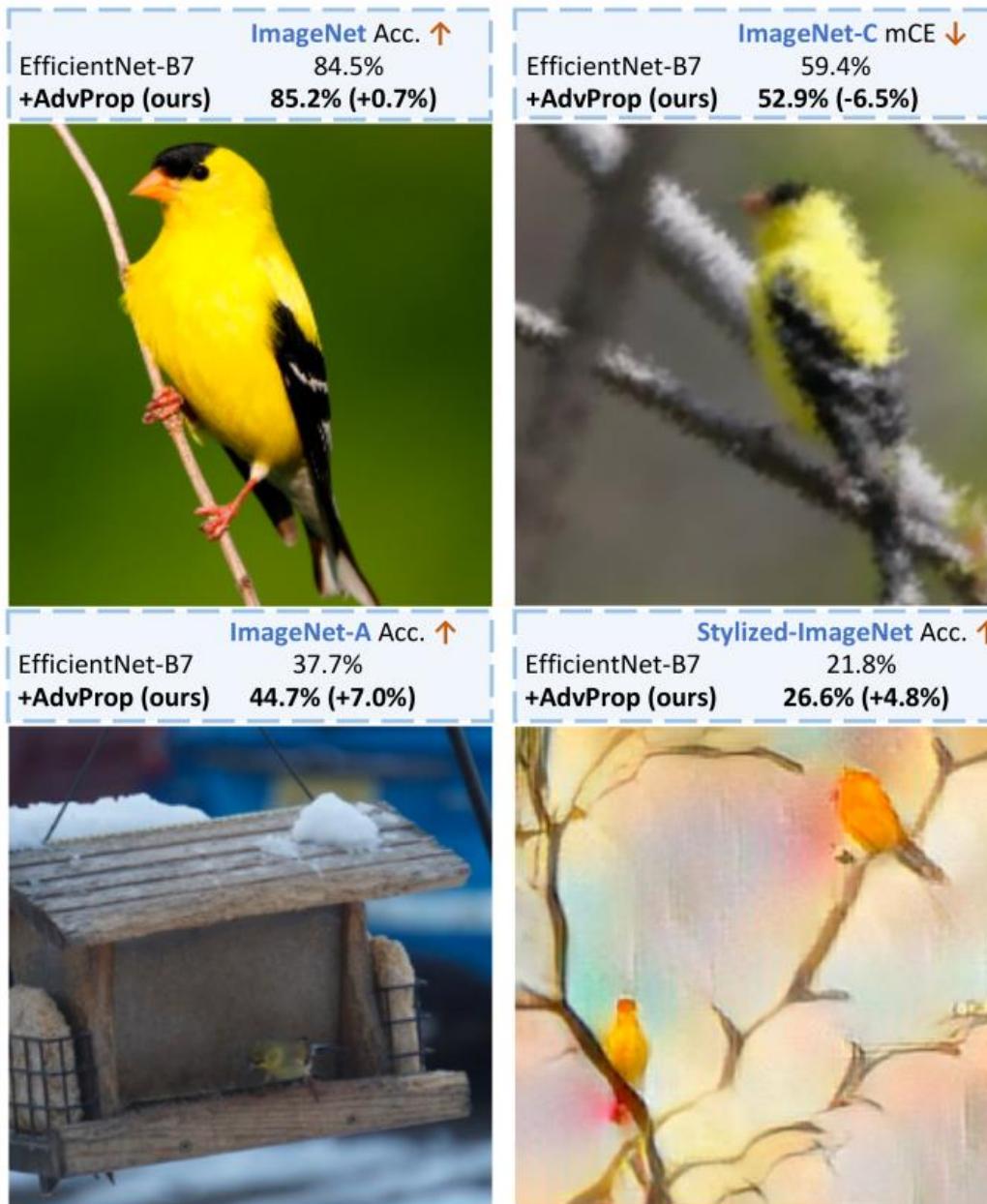


Therefore, we can see that the input image goes through network once and then objects can be detected. And we can have end-to-end learning.

Q7. Adversarial Examples Improve Image Recognition

Answer:

Adversarial examples crafted by adding imperceptible perturbations to images can lead to (ConvNets) Convolutional Neural Networks to make wrong predictions. The existence of adversarial examples not only reveal limited generalization ability of ConvNets, but also poses security threats on the real-world deployment of these models. Since the first discovery of the vulnerability of ConvNets to adversarial attacks, many efforts have been made to improve network robustness.



Above Fig.: AdvProp improves image recognition. By training model on ImageNet, AdvProp helps EfficientNet-B7 to achieve 85.2% accuracy on ImageNet, 52.9% mCE (mean corruption error, lower is better) on ImageNet-C, 44.7% accuracy on ImageNet-A and 26.6% accuracy on Stylized-ImageNet, beating its vanilla counterpart by 0.7%, 6.5%, 7.0% and 4.8%, respectively. These sample images are randomly selected from category “goldfinch.”

In this paper, rather than focusing on defending against adversarial examples, we shift our attention to leveraging adversarial examples to improve accuracy. Previous works show that training with adversarial examples can enhance model generalization but are restricted to certain situations—the improvement is only observed either on small datasets (*e.g.*, MNIST) in the fully-supervised setting [5], or on larger datasets but in the semi-supervised setting [21, 22]. Meanwhile, recent works [15, 13, 31] also suggest that training with adversarial examples on large datasets, *e.g.*, ImageNet [23], with supervised learning results in performance degradation on clean images. To summarize, it remains an open question of how adversarial examples can be used effectively to help vision models.

We observe all previous methods jointly train over clean images and adversarial examples without distinction, even though they should be drawn from different underlying distributions. We hypothesize this distribution mismatch between fresh examples and adversarial examples is a key factor that causes performance degradation in previous works.

Q8. Advancing NLP with Cognitive Language Processing Signals

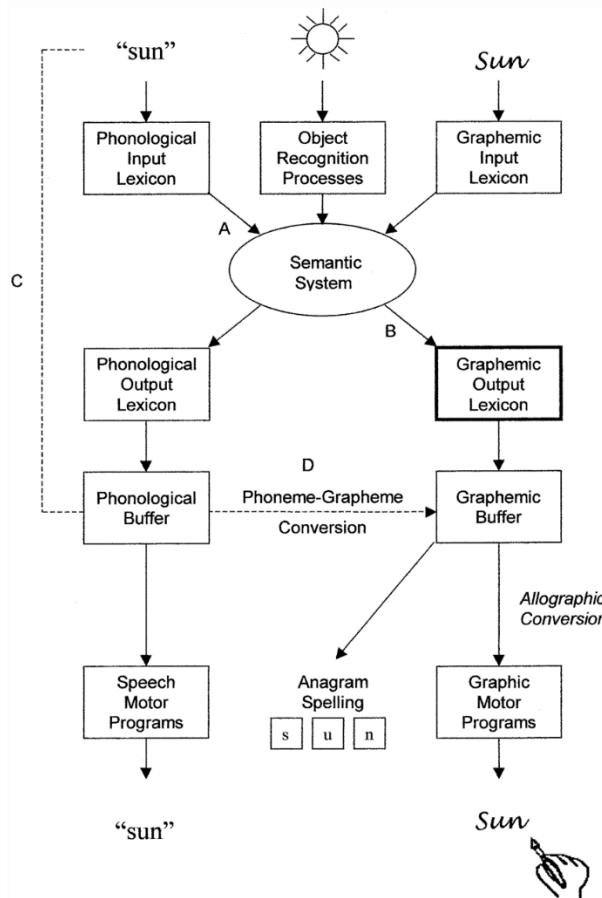
Answer:

When reading, humans process language “automatically” without reflecting on each step — Humans string words together into sentences, understand the meaning of spoken and written ideas, and process language without overthinking about how the underlying cognitive process happens. This process generates cognitive signals that could potentially facilitate natural language processing tasks.

In recent years, collecting these signals has become increasingly accessible and less expensive Papoutsaki et al. (2016); as a result, using cognitive features to improve NLP tasks has become more popular. For example, researchers have proposed a range of work that uses eye-tracking or gaze signals to improve part-of-speech tagging (Barrett et al., 2016), sentiment analysis (Mishra et al., 2017), named entity recognition Hollenstein and Zhang (2019), among other tasks. Moreover, these signals have been used successfully to regularize attention in neural networks for NLP Barrett et al. (2018).

However, most previous work leverages only eye-tracking data, presumably because it is the most accessible form of cognitive language processing signal. Also, most state-of-the-artwork(SOTA) focused on improving a single task with a single type of cognitive signal. But can cognitive processing signals bring consistent improvements across modality (*e.g.*, eye-tracking and EEG) and across various NLP

tasks? And if so, does the combination of different sources of cognitive signals bring incremental improvements?



Q8. Do you have any idea how can we use NLP on News headlines to predict index trends?

Answer:

Traders generally look up information about the company they are looking to buy shares into, for long and short trading. A frequent source of information is news media, which provides updates about the company's activities, such as expansion, better or worse revenues than expected, new products and much more. Depending on the news, trader can determine a bearish or bullish trend and decide to invest in it.

We may be able to correlate overall public sentiments towards the company and its stock price: Apple is generally well-liked by the public, receives daily news coverage of its new product and financial stability, and its stock has been growing steadily. These facts may be correlated but first may not cause the second; we will analyze if news coverage can be used to predict the market trend. To do so, we will examine the top 25 news headlines of each open-market day from 2008 to late 2015 and try to predict

the end-of-day value of DJIA index for the same day. The theory behind predicting same day value is that traders will respond to news quickly and thus, the market will adjust within an hour of release. Therefore in the single business day, if the news is spread during business hours, its effect may be measured before closing bell of the market.

The motivation behind this analysis is that humans take decision using most of the available information. This usually takes several minutes to discover new information and take the decision. An algorithm is capable of processing gigabytes of texts from multi-source streams in second. We could potentially exploit this difference in order to create a trading strategy.

NLP (Natural Language Processing) techniques can be used to extract different information from headlines such as sentiment, subjectivity, context and named entities. We obtain an indicator vector using each of these techniques, which allows us to train different algorithms to predict a trend. To predict these values, we can use several methods that should be well suited for this type of information: Linear regression, Support Vector Machine(SVM), Long Short-Term Memory(LSTM) recurrent neural network, and a dense feed-forward (MLP) neural network. We included techniques used by Bollen et al (2010), which resulted in state-of-the-art(SOTA) results. We will also analyze the method used in other studies with a similar context

Information in headlines

Latent Sentiment Analysis is done by building up a corpus of labeled words which usually connote a degree of +ve or -ve sentiment. We can extend the corpus to include emoticons (i.e. “:-)”) and expression, which often correlates to strong emotions. Naive sentiment analysis consists of a lookup of each word in sentence to be analyzed and evaluation of a score for sentence overall. This approach is limited by its known vocabulary, which can be mitigated by context analysis and introduction of synonyms. Second limitation is sarcasm, which is prevalent in twitter feed analysis. The sentiment inferred by words is opposed to the sentiment assumed by the user. This is mitigated by technique detecting sarcasm which

lead to a polarity flip of such tweets.

Sentiment analysis gives insight on how favorable the media is and maybe the bias traders may have towards buying or selling.

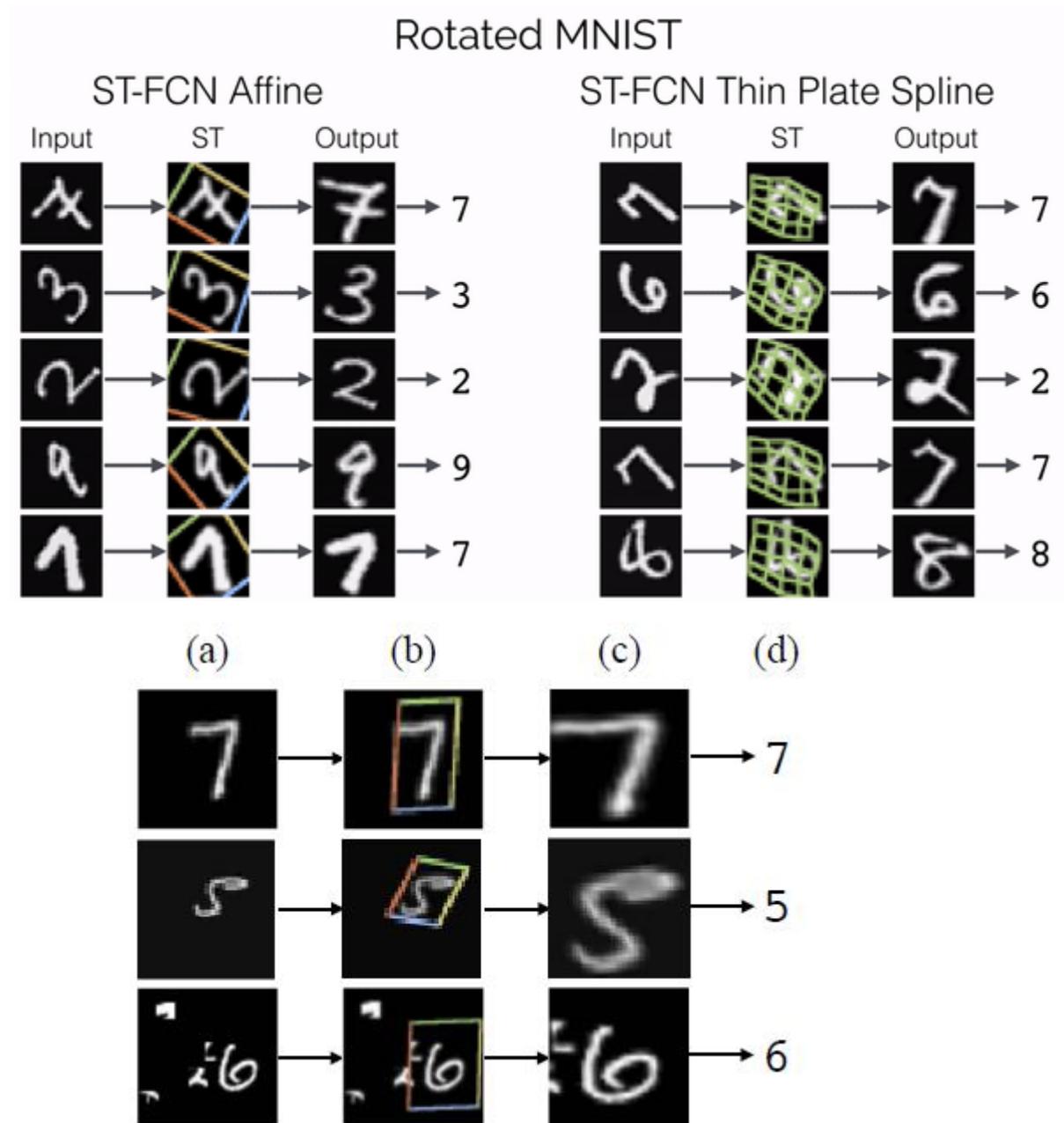
Another NLP technique which gave promising results was context analysis. This is a recent deep learning approach where you rely on a large corpus of text in order to learn and predict the words around a target. You can then deduce in what context it usually appears. The result is a vector representing each word. Other vectors with little distance are usually synonyms. The representation also allows us to do algebra, such as the famous “king - man + woman = queen”

Learning this representation offers the possibility of associating a specific context with a bullish or bearish market.

Q1.What is STN?

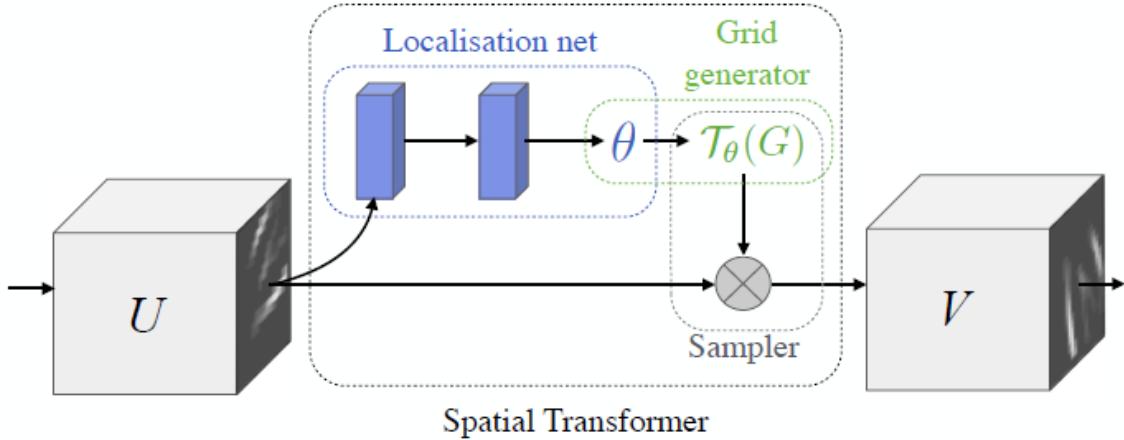
Answer:

STN stands for Spatial Transformer Network for image classification. Google Deepmind briefly reviews it. STN helps to crop out and scale-normalizes appropriate region, which can simplify the subsequent classification task and lead to better classification performance as below:



(a) Input Image with Random Translation, Scale, Rotation, and Clutter, (b) STN Applied to Input Image, (c) Output of STN, (d) Classification Prediction

Spatial Transformer Network (STN)



Source

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \mathcal{T}_\theta(G_i) = \mathbf{A}_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

Target

- STN is composed of **Localisation Net**, **Grid Generator**, and **Sampler**.

Localization Net

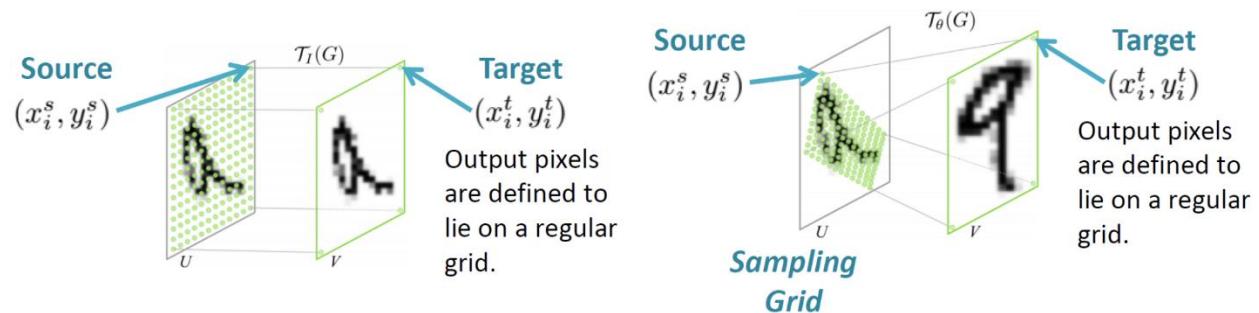
- With **input feature map U** , with (width) W , (height) H , and C channels, **outputs are θ** , parameters of transformation $T\theta$. It can be learned as affine transform as above. Or to be more constrained, such as the used for attention which only contains scaling and translation as below:

$$A_\theta = \begin{bmatrix} s & 0 & t_x \\ 0 & s & t_y \end{bmatrix}$$

Grid Generator

- Suppose we have a regular grid G , this G is a set of points with **source coordinates** (xs_i, ys_i) , which act as **input**.
- Then we **apply transformation** $T\vartheta$ on G , i.e., $T\vartheta(G)$.
- After $T\vartheta(G)$, a set of points with **destination coordinates** (xt_i, yt_i) is **outputted**. These points have been altered based on the transformation parameters. It can be Translation, Scale, Rotation or More Generic Warping depending on how we set ϑ as mentioned above.

Sampler



- Based on the new set of coordinates (xt_i, yt_i) , we generate a **transformed output feature map** V . This V is translated, scaled, rotated, warped, projective transformed or affined, whatever.
- It is noted that STN can be applied to not only input image but also intermediate feature maps.

Q2.What is decaNLP?

Answer:

We introduced the Natural Language Decathlon (decaNLP) to explore models that generalize to many different kinds of Natural Language Processing(NLP) tasks. decaNLP encourages single model to

simultaneously optimize for 10 tasks: question answering, machine translation, document summarization, semantic parsing, sentiment analysis, natural language inference(NLI), semantic role labeling, relation extraction, goal-oriented dialogue, and pronoun resolution.

We frame all the tasks as question answering [Kumar et al., 2016] by allowing task specification to take the form of a natural language question q : all inputs have a context, question, and answer (Fig. 1). Traditionally, NLP examples have inputs x and output y , and the underlying task t is provided through explicit modeling constraints. Meta-learning approaches include t as additional input. Our approach does not use the single representation for any t but instead uses natural language questions that describe underlying tasks. This allows single models to multitask effectively and makes them more suitable as pre-trained models for transfer learning and meta-learning: natural language questions allow a model to generalize to entirely new tasks through different but related task descriptions.

The MQAN (multitask question answering network) is designed for decaNLP and makes use of a novel dual attention and multi-pointer-generator decoder to multitask across all tasks in decaNLP. Our results represent that training the MQAN jointly on all tasks with the right anti-curriculum strategy can achieve performance comparable to that of ten separate MQANs, each trained separately. An MQAN pretrained on decaNLP shows improvements in transfer learning for machine translation and named entity recognition(NER), domain adaptation for sentiment analysis and natural language inference(NLI), and zero-shot capabilities for text classification. Though not explicitly designed for any one job, MQAN proves to be a robust model in a single-task setting as well, achieving state-of-the-art results on the semantic parsing component of decaNLP.

Examples

Question	Context	Answer	Question	Context	Answer
What is a major importance of Southern California in relation to California and the US?	...Southern California is a major economic center for the state of California and the US....	major economic center	What has something experienced?	Areas of the Baltic that have experienced eutrophication .	eutrophication
What is the translation from English to German?	Most of the planet is ocean water.	Der Großteil der Erde ist Meerwasser	Who is the illustrator of Cycle of the Werewolf?	Cycle of the Werewolf is a short novel by Stephen King, featuring illustrations by comic book artist Bernie Wrightson .	Bernie Wrightson
What is the summary?	Harry Potter star Daniel Radcliffe gains access to a reported £320 million fortune...	Harry Potter star Daniel Radcliffe gets £320M fortune...	What is the change in dialogue state?	Are there any Eritrean restaurants in town?	food: Eritrean
Hypothesis: Product and geography are what make cream skimming work. Entailment , neutral, or contradiction?	Premise: Conceptually cream skimming has two basic dimensions – product and geography.	Entailment	What is the translation from English to SQL?	The table has column names... Tell me what the notes are for South Australia	SELECT notes from table WHERE 'Current Slogan' = 'South Australia'
Is this sentence positive or negative?	A stirring, funny and finally transporting re-imagining of Beauty and the Beast and 1930s horror film.	positive	Who had given help? Susan or Joan ?	Joan made sure to thank Susan for all the help she had given.	Susan

In the above figure: Overview of the decaNLP dataset with one example from each decaNLP task in the order presented in Section 2. They show how the datasets were pre-processed to become question

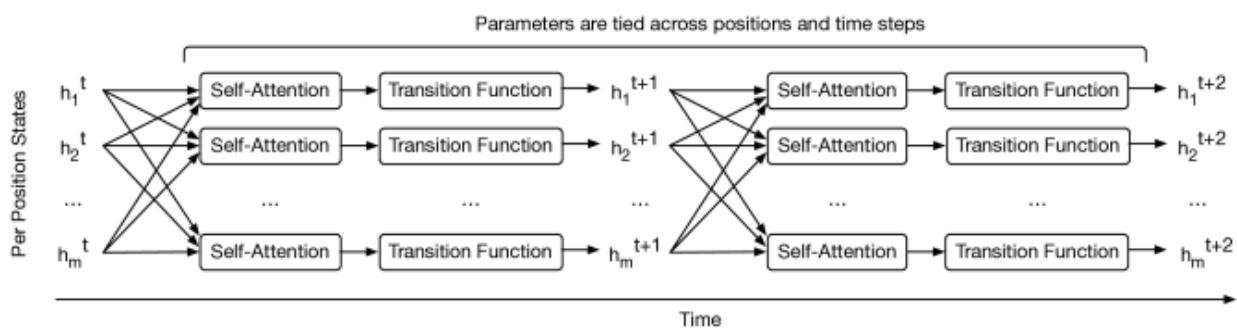
answering problems. Answer words in red are generated by pointing to the context, in green from the issue, and in blue if they are made from a classifier over the output vocabulary.

Q3.Universal Transformers

Answer:

Convolutional and fully-attentional feed-forward architectures such as the Transformer model have recently emerged as viable alternatives to RNNs(Recurrent neural networks) (for the range of sequence modeling tasks, notably machine translation ([JonasFaceNet2017](#); [transformer](#),). These architectures address the significant shortcoming of RNNs, namely their inherently sequential computation, which prevents parallelization across elements of input sequence while still addressing vanishing gradients problem ([vanishing-exploding-gradient](#)). Transformer model, in particular, achieves this by relying entirely on the self-attention mechanism ([decomposableAttnModel](#); [lin2017structured](#)) to compute series of context-informed vector-space representations of symbols in its input and output, which are then used to predict distributions over subsequent symbols as the model predicts output sequence symbol-by-symbol. Not only in this mechanism straightforward to parallelize, but as each symbol's representation is also directly informed by all other symbols representations, this results in an active global receptive field. This stands, in contrast, to, e.g., convolutional architecture, which typically has limited receptive field.

Notably, however, Transformer foregoes the (Recurrent Neural Network)RNN's inductive bias towards learning recursive or iterative transformations. Our experiments indicate that this inductive bias may be important for several algorithmic and language understanding tasks of varying complexity: in contrast to models such as the Neural Turing Machine, the Neural GPU, or Stack RNNs, the Transformer does not generalize well to input lengths not encountered during training.



In this paper, we propose a *Universal Transformer*. It combines the parallelizability and global receptive field of a Transformer model with the recurrent inductive bias of RNNs, which seems to be better suited to range of algorithmic and natural language understanding(NLU) sequence-to-sequence problems. As the name implies, in contrast to standard Transformer, under certain assumptions, a Universal Transformer can be shown to be computationally universal.

In each step, the Universal Transformer iteratively refine its representations for all positions in sequence in parallel with self-attention mechanism decomposableAttnModel (); lin2017structured (), followed by the recurrent transformation consisting of a depth-wise separable convolution (xception2016) or a position-wise fully-connected layer (see above Fig). We also extended the Universal Transformer by employing an adaptive computation time mechanism at each position in sequence (graves2016adaptive), allowing model to choose the required number of refinement steps for each symbol dynamically.

When running for fixed number of steps, the Universal Transformer is equivalent to a multi-layer Transformer with a tied parameter across its layers. However, another, and possibly more informative, way of characterizing Universal Transformer is as recurrent function evolving per-symbol hidden states in parallel, based at each step on a sequence of the previous unknown state. In this way, it is similar to architectures such as Neural GPU and the Neural Turing Machine. The Universal Transformer thereby retains the attractive computational efficiency of original feed-forward Transformer model, but with an added recurrent inductive bias of RNNs. In its adaptive form, we show that the Universal Transformer can effectively interpolate between the feed-forward, fixed-depth Transformer, and a gated, recurrent architecture running for several steps depending on the input data.

Our experimental results show that its recurrence improve results in machine translation, where Universal Transformer outperforms the standard Transformer with a same no.of parameters. In experiments on several algorithmic tasks, Universal Transformer consistently improves significantly over LSTM(Long Short Term Memory) RNNs and the standard Transformer. Furthermore, on bAbI and LAMBADA text understanding data sets, the Universal Transformer achieves a new state of the art.

Q4. What is StarSpace in NLP?

Answer:

We introduce StarSpace, the neural embedding model that is general enough to solve a wide variety of problems:

- Other labeling tasks, or Text classification, e.g., sentiment classification.

- Ranking of the set of entities, e.g., a classification of web documents given a query.
- Collaborative filtering-based recommendation, e.g., recommending documents, videos or music.
- Content-based recommendation where content is defined with discrete features, e.g., words of documents.
- Embedding graphs, e.g., multi-relational graphs such as Freebase.
- Learning word, sentence, or document embeddings.

It can be viewed as a straight-forward and efficient strong baseline for any of these tasks. In experiment, it is shown to be on par with or outperforming several competing methods while being generally applicable to cases where many of that method are not.

The method works by learning entity embeddings with discrete feature representation from relations among collections of those entities directly for the task of ranking or classification of interest. In the general case, StarSpace embeds objects of different types into a vectorial embedding space; hence, the “star” (“*,” meaning all types) and “space” in a name and in that familiar space compares them against each other. It learns to rank the set of entities, documents, or objects given a query entity, document, or object, where the query is not necessarily of the same type as the items in the set.

Q5. TransferTransfo in NLP

Answer:

Non-goal-oriented dialogue systems (chatbots) are interesting test-bed for interactive Natural Language Processing (NLP) systems and are also directly useful in wide range of applications ranging from technical support services to entertainment. However, building intelligent conversational agent remains an unsolved problem in artificial intelligence(AI) research. Recently, recurrent neural network(RNN) based models with sufficient capacity and access to large datasets attracted large interest when first attempted. It showed that they were capable of generating meaningful responses in some chit-chat settings. Still, further inquiries in the capabilities of these neural network

architectures and developments indicated that they were limited which made communicating with them a rather unsatisfying experience for human beings.

The main issues with these architectures can be summarized as:

- (i) the wildly inconsistent outputs and the lack of a consistent personality (Li and Jurafsky, [2016](#)),
- (ii) the absence of long-term memory as these models have difficulties in taking into account more than the last dialogue utterance; and
- (iii) a tendency to produce consensual and generic responses that are vague and not engaging for humans (Li, Monroe, and Jurafsky, [2016](#)).

In this work, we take a step toward more consistent and relevant data-driven conversational agents by proposing a model architecture, associated training and generation algorithms which are able to significantly improve over the traditional seq-2-seq and information-retrieval baselines in terms of (i) relevance of the answer (ii) coherence with a predefined personality and dialog history, and (iii) grammaticality and fluency as evaluated by auto.

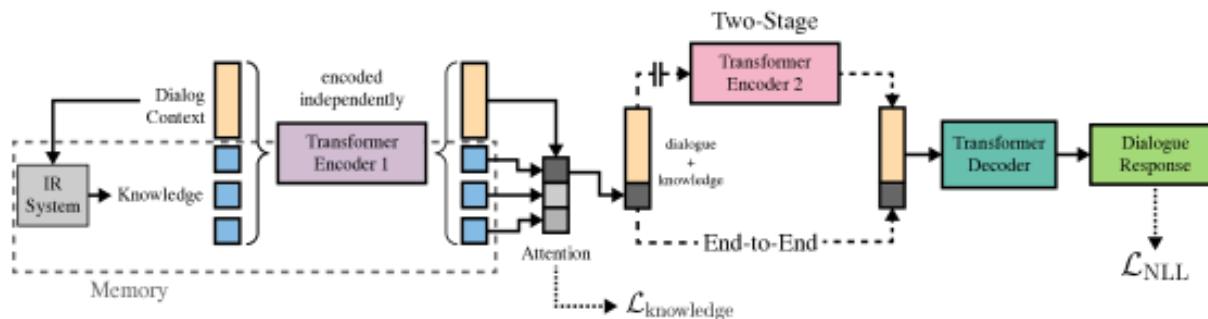
Q6. Wizard of Wikipedia: Knowledge-Powered Conversational Agents

Answer:

Arguably, one of the critical goals of AI and the ultimate goal of natural language research is for the human to be able to talk to the machine. In order to get close to this goal, machines must master the no. of skills: to be able to comprehend language, employ memory to retain and recall knowledge, to reason about these concept together, and finally output a response that both fulfills functional goals in conversation while simultaneously being captivating to their human speaking partner. The current state-of-the-art(SOTA) approaches, sequence to sequence(seq2seq) models of various kinds (Sutskever et al., [2014](#); Vinyals & Le, [2015](#); Serban et al., [2016](#); Vaswani et al., [2017](#)) attempt to address some of these skills, but generally suffer from inability to bring memory and knowledge to bear; as indicated by their name, they involve encoding input sequence, providing limited reasoning by transforming their hidden state given input, and then decoding to the output. To converse intelligently on the given topic, the speaker needs knowledge of that subject, and it is our contention here that more direct knowledge memory mechanisms need to be employed. In this work, we consider setups where this can be naturally measured and built.

We consider the task of open-domain dialogue, where two speakers conduct open-ended chit-chat given an initial starting topic, and during the conversation, the topic can broaden or focus on related themes. During such conversations, an interlocutor can glean new information and personal points of view from

their speaking partner, while providing themselves similarly. This is a challenging task as it requires several components not found in many standard models. We design a set of architectures specifically for this goal that combine elements of Memory Network architectures (Sukhbaatar et al., 2015) to retrieve knowledge and read and condition on it, and Transformer architectures (Vaswani et al., 2017) to provide state-of-the-art text representations and sequence models for generating outputs, which we term Transformer Memory Networks.



Q7. ERASER: A Benchmark to Evaluate Rationalized NLP Models

Answer:

Movie Reviews

In this movie, ... Plots to take over the world. The acting is great! The soundtrack is run-of-the-mill, but the action more than makes up for it

(a) Positive (b) Negative

e-SNLI

H A man in an orange vest leans over a pickup truck
P A man is touching a truck

(a) Entailment (b) Contradiction (c) Neutral

Commonsense Explanations (CoS-E)

Where do you find the most amount of leafs?

(a) Compost pile (b) Flowers (c) Forest (d) Field (e) Ground

Evidence Inference

Article Patients for this trial were recruited ... Compared with 0.9% saline, 120 mg of inhaled nebulized furosemide had no effect on breathlessness during exercise.

Prompt With respect to breathlessness, what is the reported difference between patients receiving placebo and those receiving furosemide?

(a) Sig. decreased (b) No sig. difference (c) Sig. increased

Interest has recently grown in interpretable (Natural Language Processing) NLP systems that can reveal **how** and **why** model make their predictions. But work in this direction has been conducted on the

different dataset with correspondingly different metrics, and inherent subjectivity in defining what constitute ‘interpretability’ has translated into researcher using different metrics to quantify performance. We aimed to facilitate measurable progress on designing interpretable NLP(Natural Language Processing) models by releasing the standardized benchmark of datasets — augmented and repurposed from pre-existing corpora, and spanning the range of NLP tasks — and associated metrics for measuring the quality of rationales. We refer to this as ERASER(Evaluating Rationales And Simple English Reasoning) benchmark.

In curating and releasing ERASER we take inspiration from stickiness of GLUE (Wang et al., 2019b) and SuperGLUE Wang et al. (2019a) benchmarks for evaluating progress in natural language understanding(NLU) tasks. These have enabled rapid growth in models for inclusive language representation learning. We believe still somewhat nascent subfield of interpretable NLP(Natural Language Processing) stands to similarly benefit from the analogous collection of standardized datasets or tasks and metric.

‘Interpretability’ is the broad topic with many possible realizations Doshi-Velez and Kim ([2017](#)); Lipton ([2016](#)). In ERASER, we focuses specifically on *rationales*, i.e., snippets of text from the source document that support a specific categorization. All datasets contained in ERASER include such rational, explicitly marked by annotators as supporting specific classifications. By definition, rationales should be *sufficient* to categorize document, but they may not be comprehensive. Therefore, for some dataset, we have collected *complete* rationales, i.e., in which *all* evidence supporting the classification has been marked.

How one measures ‘quality’ of extracted rationales will invariably depend on their intended use. With this in mind, we propose the suite of metrics to evaluate rationales that might be appropriate for different scenarios. Widely, this includes measures of agreement with human-provided rationales and assessment of *faithfulness*. The latter aim to capture extent to which rationales provided by the model, in fact, informed its prediction.

While we propose metrics that we think are reasonable, we view a problem of designing metrics for evaluating rationales—especially for capturing faithfulness — as a topic for further research that we hope that ERASER will help facilitate. We plan to revisit metrics proposed here in future iterations of benchmark, ideally with input from community. Notably, while we provide a ‘leaderboard,’ this is perhaps better viewed as the ‘results board’; we do not privilege any particular metric. Instead, we hope that ERASER permits comparison between models that provide rationales wrt different criteria of interest.

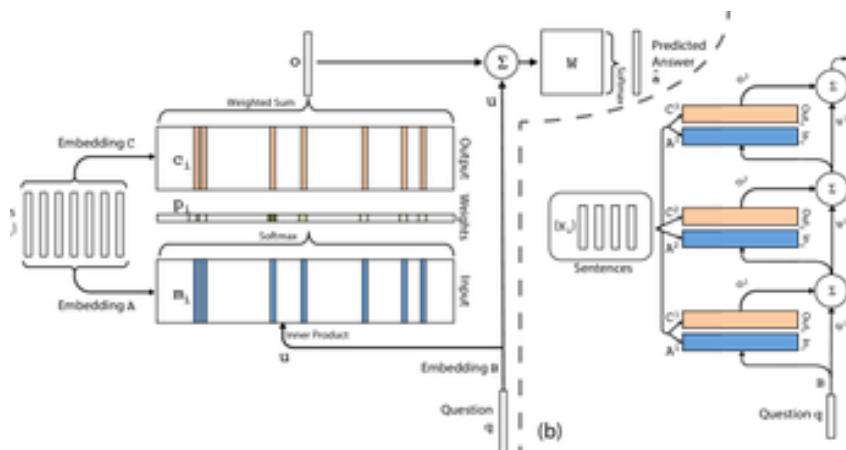
Q8. End to End memory networks

Answer:

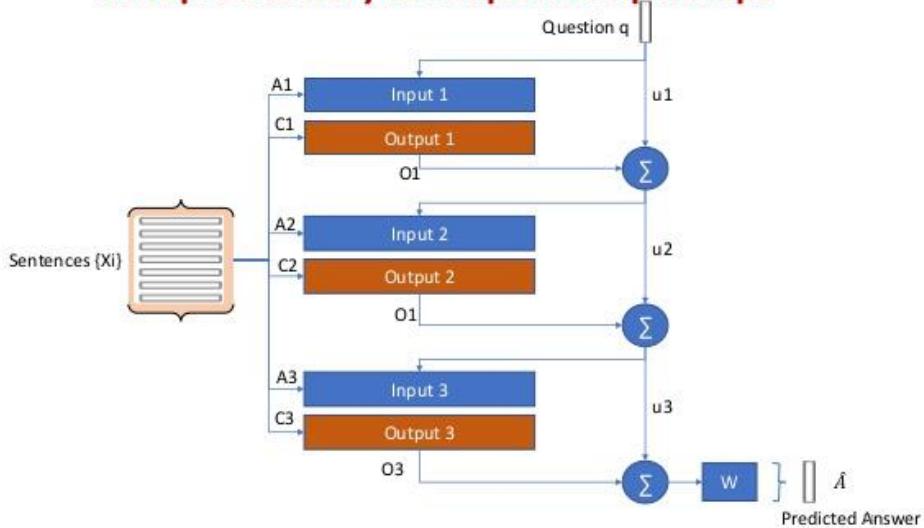
Two grand challenges in artificial intelligence(AI) research have been to build a model that can make multiple computational step in the service of answering the question or completing the task, and models that can describe long term dependencies in sequential data.

Recently there has been the resurgence in models of computation using explicit storage and a notion of attention; manipulating such storage offers an approach to both of these challenges. In, the storage is endowed with continuous representation; reads from and writes to storage, as well as other processing steps, are modeled by actions of neural networks.

In this work, we present the new recurrent neural network (RNN) architecture where recurrence reads from possibly large external memory multiple times before outputting symbol. Our model can be considered the continuous form of the Memory Network implemented in. The model in that work was not easy to train via back-propagation and required supervision at each layer of a network. The continuity of model we present here means that it can be trained end-to-end from input-output pairs, and so applies to more tasks, i.e., tasks where such supervision is not available, like in language modeling or realistically supervised question answering tasks. Our model can also be seen as version of RNNsearch with multiple computational steps per output symbol. We will show experimentally that various hops over the long-term memory are crucial to excellent performance of our model on these tasks, and that training the memory representation can be integrated in a scalable manner into our end-to-end neural network model.



Multiple Memory Lookups: Multiple Hops



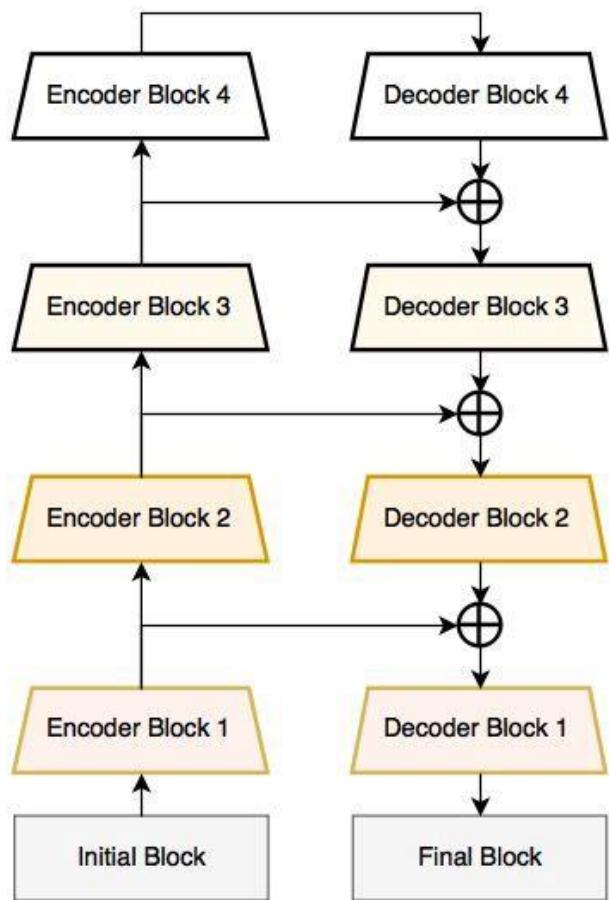
Q9. What is LinkNet?

Answer:

From my experience, LinkNet is lightning fast, which is one of the main improvements the authors site in their summary. LinkNet is a relatively lightweight network with around 11.5 million parameters; networks like VGG have more than 10x that amount.

The structure of LinkNet is to use a series of encoder and decoder blocks to break down the image and build it back up before passing it through a few final convolutional layers. The structure of the network was designed to minimize the number of parameters so that segmentation could be done in real-time.

I performed some tests of the LinkNet architecture but did not spend too much time iterating to improving the models.

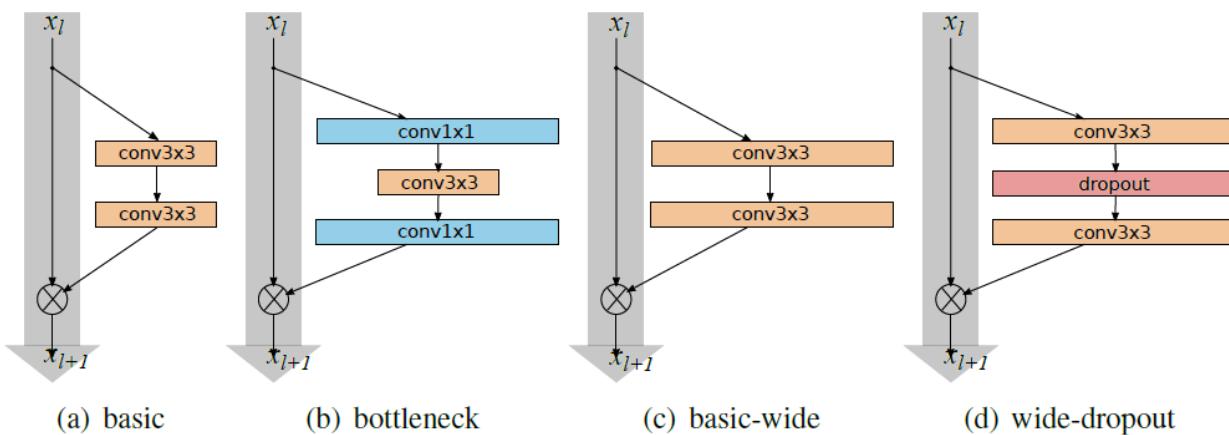


Q1. What is WRN?

Answer:

WRN: It stands for Wide Residual Networks is presented. By widening Residual Network (ResNet), the network can be more superficial or shallow with same accuracy or improved accuracy. More external network means:

- the number of layers can be reduced.
- Training time can be shorter, as well.



Problems on Residual Network (ResNet)

Circuit Complexity Theory

The authors of residual networks(ResNet) tried to make them as thin as possible in favor of increasing their depth and having less parameters and even introduced a «bottleneck» block, which makes ResNet blocks even thinner.

Diminishing Feature Reuse

However, As gradient flows through network, there is nothing to force it to go through residual block weights, and it can avoid learning anything during training, so there may be either only few blocks that learn useful representations or many blocks share very little information with a small contribution to the final goal. This problem was formulated as a diminishing feature reuse.

WRNs (Wide Residual Networks)

In WRNs, plenty of parameters are tested like the design of ResNet block, how deep (deepening factor λ), and how extensive (widening factor k) within the ResNet block.

When $k=1$, it has the same width as the *ResNet*. While $k>1$, it is k time wider than *ResNet*.

WRN- $d\cdot k$ means the WRN has a depth of d and with widening factor k .

- *Pre-Activation ResNet* is used in CIFAR-10, CIFAR-100, and SVHN datasets. Original *ResNet* is used in the ImageNet dataset.
- The significant difference is that *Pre-Activation ResNet* has the structure of performing batch norm and ReLU before convolution (i.e., BN-ReLU-Conv) while original *ResNet* has the structure of Conv-BN-ReLU. And *Pre-Activation ResNet* is generally better than the original one, but it has no visible improvement in ImageNet when layers are only around 100.

The design of the ResNet block

block type	depth	# params	time,s	CIFAR-10
$B(1,3,1)$	40	1.4M	85.8	6.06
$B(3,1)$	40	1.2M	67.5	5.78
$B(1,3)$	40	1.3M	72.2	6.42
$B(3,1,1)$	40	1.3M	82.2	5.86
$B(3,3)$	28	1.5M	67.5	5.73
$B(3,1,3)$	22	1.1M	59.9	5.78

-
- **B(3;3):** Original «basic» block, in the above figure a.
- **B(3;1;3):** With one extra (1×1) layer in between the two 3×3 layers
- **B(1;3;1):** With the same dimensionality of all convolutions, bottleneck
- **B(1;3):** The network has the alternating (1×1 , 3×3) convolutions.
- **B(3;1):** The network has the alternating(3×3 , 1×1) convolutions.
- **B(3;1;1):** This is Network in Network style block.

B(3;3) has the smallest error rate (5.73%).

Note: The Number of depths (layers) is different is to keep the number of parameters close to each other.

Q2.What is SIMCO: SIMilarity-based object Counting?

Answer:

Most approaches for counting similar objects in images assume a single object class; when is not, ad-hoc learning is necessary. None of them are genuinely agnostic and multi-class, i.e., able to capture repeated patterns of different types without any tuning. Counting approaches are based on density or regression estimation; here, we focus on counting by detection, so the counted objects are individually detected first.

Research on agnostic counting is vital in many fields. It serves for obvious question answering, where counting questions could be made on too-specific entities outside the semantic span of the available classes (e.g., “What is the most occurrent thing?” in below Fig.). In representation learning, unsupervised counting of visual primitives (i.e., visible “things”) is crucial to obtain a rich image representation. Counting is a hot topic in cognitive robotics, where autonomous agents learn by separating sensory input into the finite number of classes (without a precise semantics), building the classification system that

counts on each of them.

Application-wise, agnostic counting may help the manual tagging of training images, providing a starting guess for the annotator on single- or multi-spectral images. Inpainting filters may benefit from a magic wand capturing repeated instances to remove.

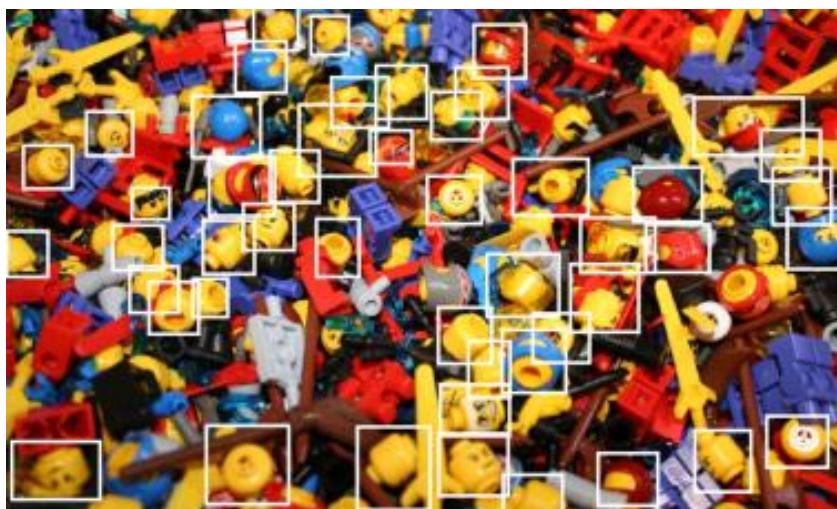


Figure: SIMCO on obvious question answering: the most occurrent object? SIMCO finds 47 LEGO heads.

In this paper, we present the SIMCO (SIMilarity-based object COnting) approach, which is entirely agnostic, i.e., with no need for any *ad-hoc* class-specific fine-tuning, and multi-class, i.e., finding different types of repeated patterns. Two main ideas characterize SIMCO.

First, every object to be counted is considered as a specialization of a basic 2D shape: this is particularly true with many and small objects (see in above Fig: LEGO heads can be approximated as circles). SIMCO incorporates this idea building upon the novel Mask-RCNN-based classifier, fine-tuned just once on a novel synthetic shape dataset, *InShape*.

The second idea is that leveraging on the 2D shape approximation of objects; one can naturally perform unsupervised grouping of the detected objects (grouping circles with circles, etc.), discovering *different* types of repeated entities (without resorting to a particular set of classes). SIMCO realizes this with a head branch in the network architecture implementing triplet losses, which provides a 64-dim embedding that maps objects close if they share the same shape class plus some appearance attributes. Affinity propagation clustering finds groups over this embedding.

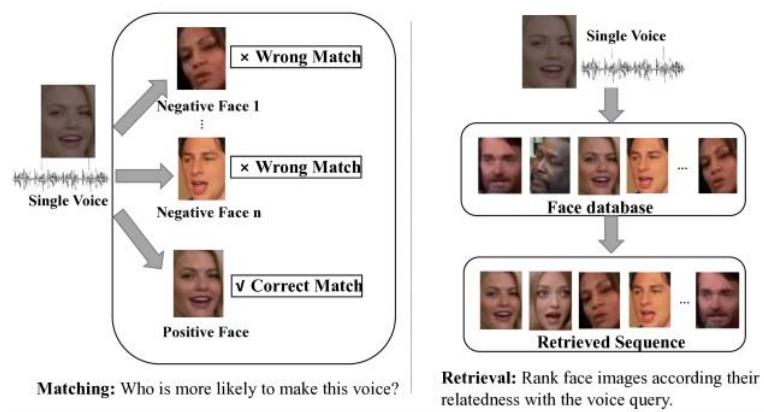
Q3. What is Voice-Face Cross-modal Matching and Retrieval?

Answer:

Studies in biology and neuroscience have shown that human's appearances are associated with their voices. Both the facial features and voice-controlling organs of individuals are affected by hormones and genetic information. Human beings can recognize this association. For example: when hearing from the phone call, we can guess the gender, the approximate age of the person on the other end of the line. When watching an unvoiced TV show. We can imagine an approximate voice by observing the face movement of the protagonist. With the recent advances of deep learning, face recognition models, and speaker recognition models have achieved exceptionally high precision. Can the associations between voices and faces be discovered algorithmically by machines? The research on this problem can benefit a lot of applications such as synchronizing video faces and talking sound, generating faces according to voices.

In recent years, much research attention has been paid on the voice-face cross-modal learning tasks, which have shown the feasibility of recognizing voice-face associations. This problem is generally formulated as a voice-face matching task and the voice-face retrieval task, as shown in Figure 1. Given a set of voice audios and faces, voice-face matching is to tell which look makes the voice when machine hearing voice audio. Voice-face retrieval is to present a sorted sequence of faces in the order of the

estimated match from a query of voice recording.



SVHF is the prior of voice-face cross-modal learning, which studies the performance of CNN-based deep network on this problem. The human's baseline for the voice-face matching task is also proposed in the paper. Both the "voice to face" and the "face to voice" matching tasks are studied in the Pins and Horiguchi's work, which exhibits similar performance on these two tasks. The curriculum learning schedule is introduced in Pins for hard negative mining. Various visualizations of the embedding vectors are presented to show the learned audio-visual associations in Kim's work. DIMNet learns the common representations for faces and voices by leveraging their relationship with some covariates such as gender and nationality. DIMNet obtains an accuracy of 84.12% on the 1:2 matching, which exceeds the human level.

Research on this problem is still in the early stage. Datasets used by previous research are always tiny, which can't evaluate the generalization ability of models sufficiently. Traditional test schemes based on random tuple mining tend to have low confidence. The benchmark for this problem needs to be established. This paper presents the voice-face cross-modal matching and retrieval framework, a dataset from Chinese speakers and a data collection tool. In the frame, cross-modal embeddings are learned with CNN-based networks, and triplet loss in a voice anchored metric space with L2-Norm constraint. An identity-based example sampling method is adopted to improve the model efficiency. The proposed framework achieves state-of-the-art performance on multiple tasks. For example, the result of 1:2 matching tested on 10 million triplets (thousands of people) reached 84.48%, which is also higher than DIMNet tested on 189 people. We have evaluated the various modules of the CNN-based framework and provided our recommendations. Even matching and retrieval based on the average of multiple voices and multiple faces are also attempted, which can further improve the performance. This task is the simplest way of analyzing video data. Large-scale datasets are used in this problem to ensure the generalization ability required in a real application. The cross-language transfer capability of the model is studied on the voice-face dataset of Chinese speakers we constructed. The series of performance