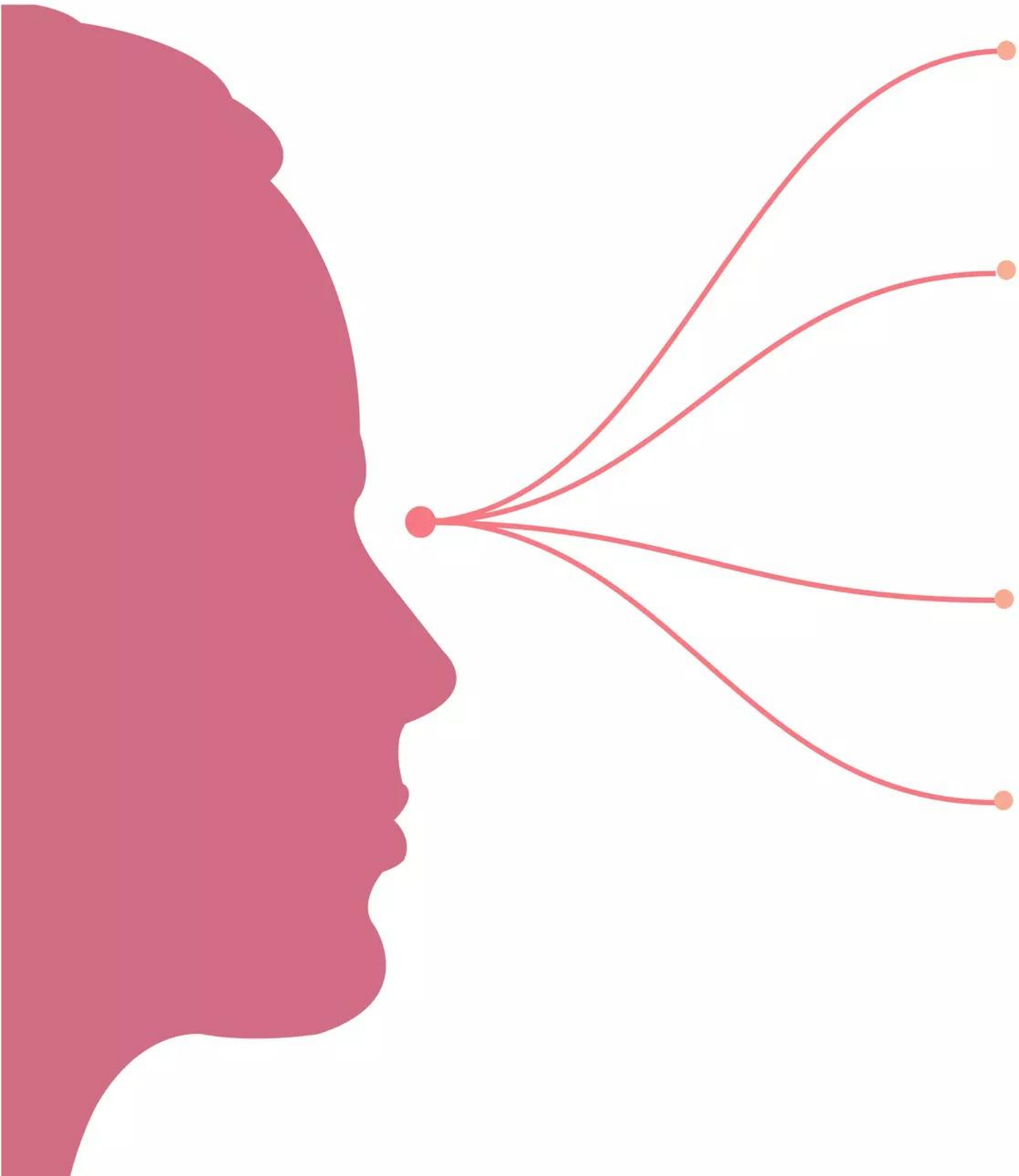
Fraud Detection System using Deep Neural Networks

Hendri Karisma

Fraudulent Transaction



Payment Fraud (phising, account take over, carding)

System abuse (promo, content, account, logistic and payment mothods especially **COD**)

Fraud not only result in financial losses but also produce some reputational risk.

Some security measures taken by bank or another multinational finance service.

[E. Duman et al, 2013]

State of the art

- There are several research in fraud detection area using some methods :
  - GASS 82.78%-91% and MBO algorithm 91.3%-94.35%
  - ANN 91.74%
  - SVM 83.06%

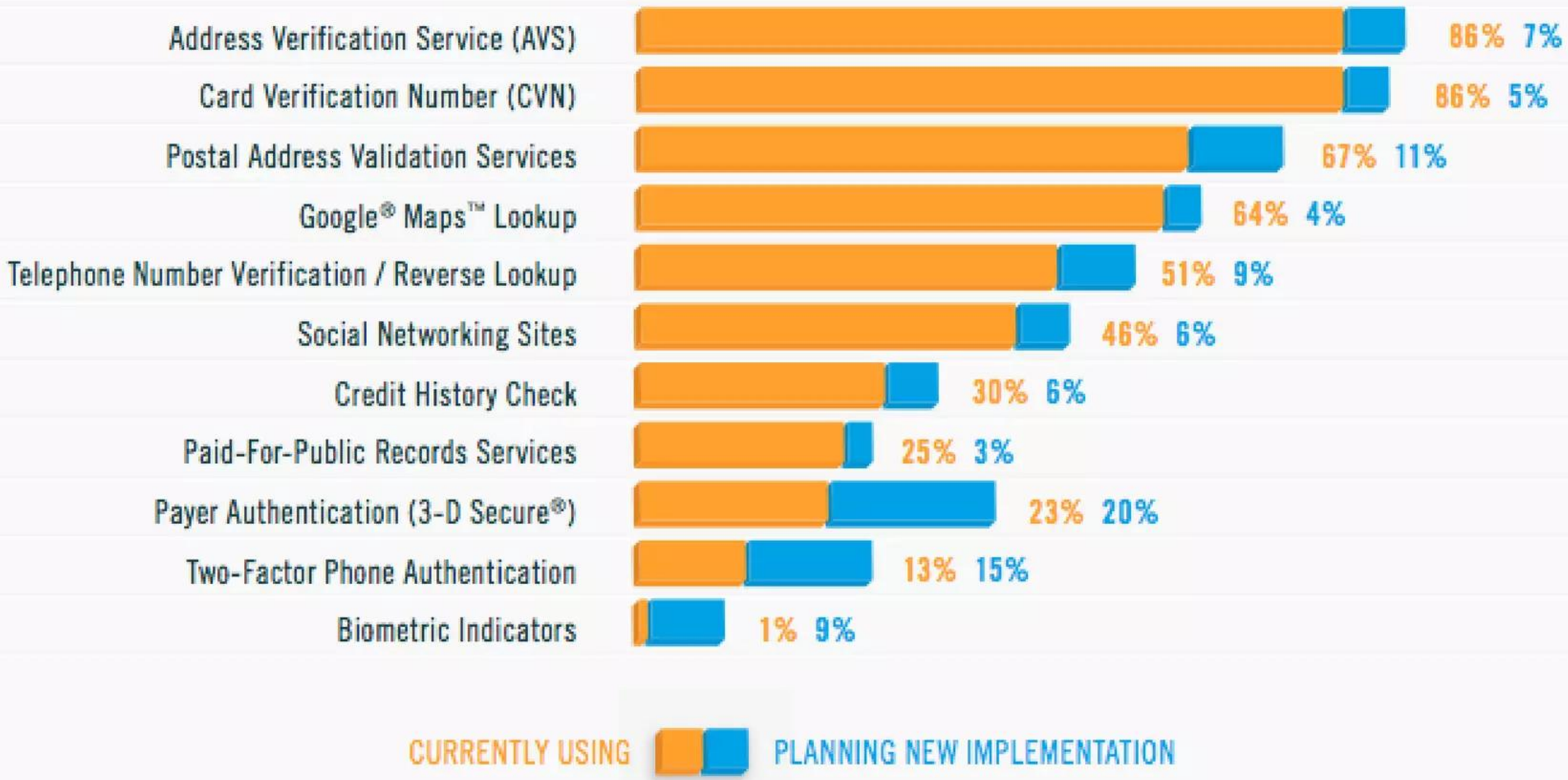
[E. Duman et al, 2013]

  - Copula-based method, extreme outlier elimination, PCA, naïve bayes , regression logistic, k-nearest neighbors, etc.

Annual Reports Cybersource

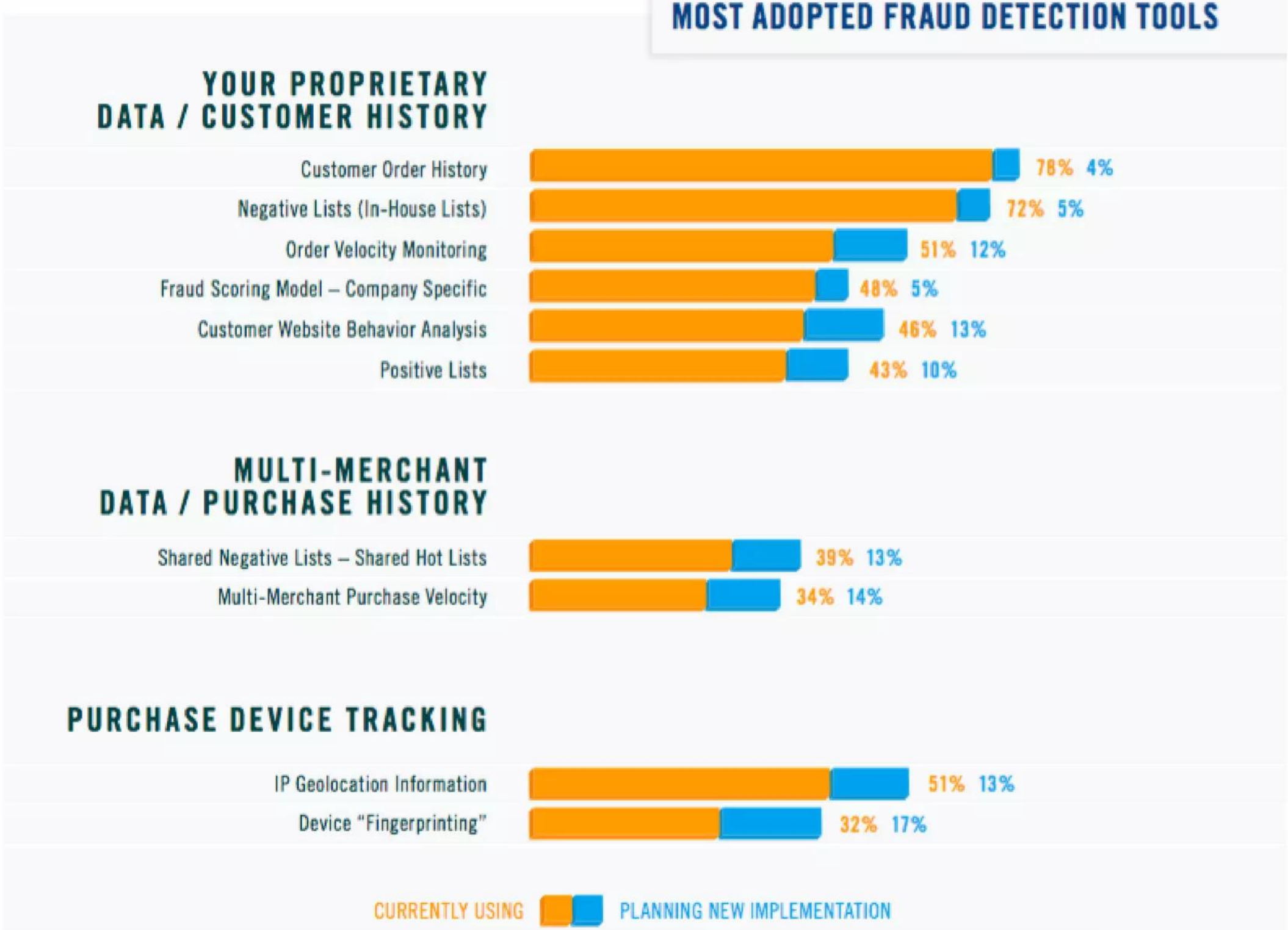
MOST ADOPTED FRAUD DETECTION TOOLS

VALIDATION SERVICES

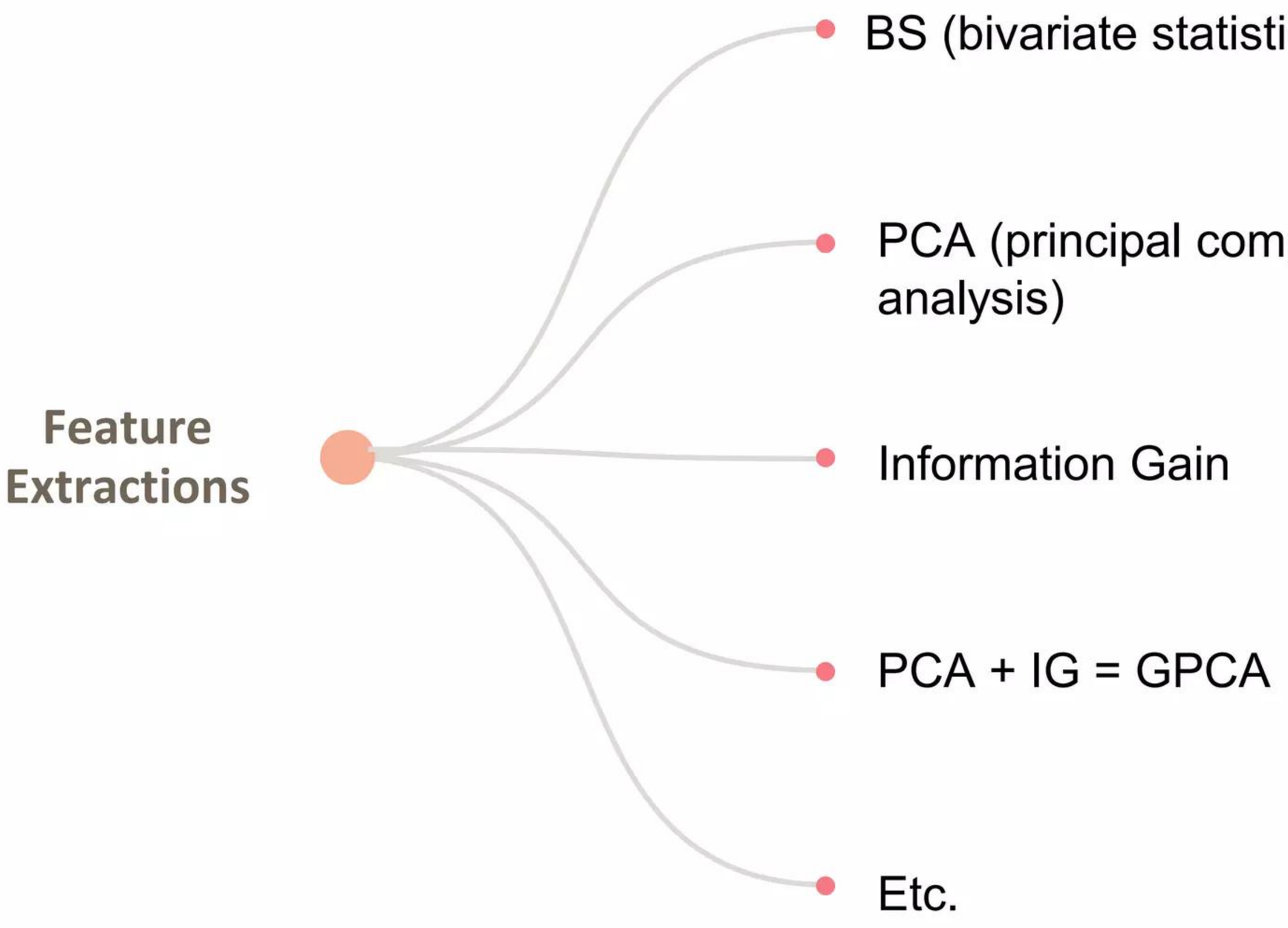


Annual Reports Cybersource

MOST ADOPTED FRAUD DETECTION TOOLS



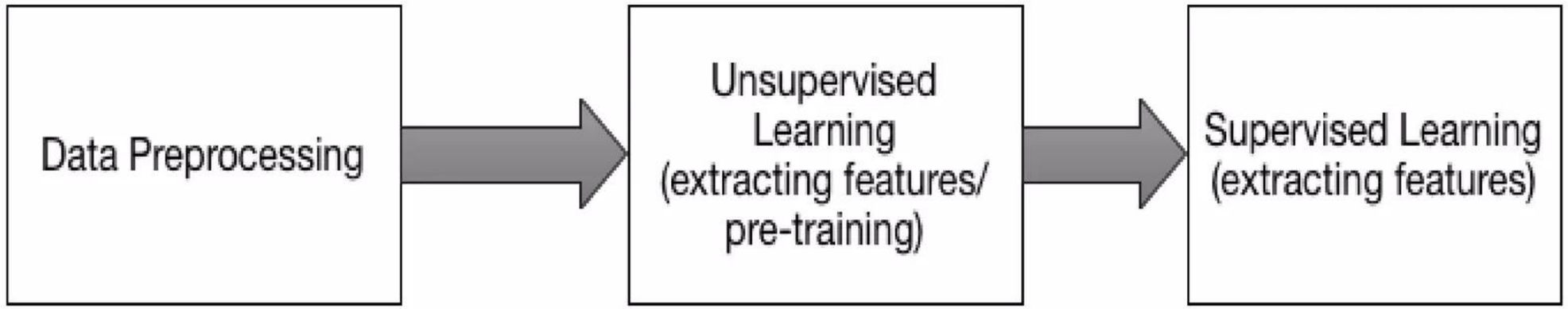
State of the art



Why Deep Learning

- Dataset is high nonlinearity
- Amount of data
- A lot of features
- Mostly unlabeled data

Deep neural networks



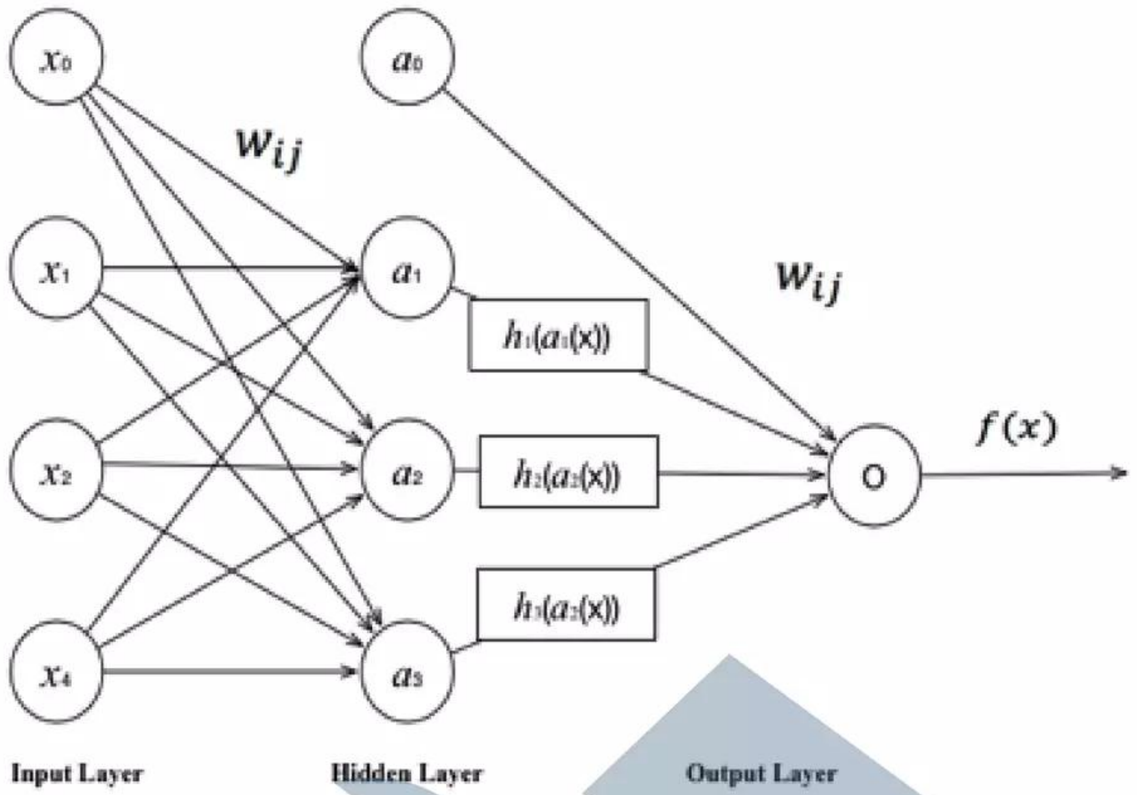
Standard neural networks

$$a(x) = b + \sum_i w_i x_i = b + W^{\mathrm{T}} x$$

$$h(x) = \varphi\big(a(x)\big) = \varphi(b + \sum_i w_i x_i)$$

$$\varphi\big(a(x)\big) = \frac{1}{1 + e^{-a(x)}}$$

$$f(x) = o(b^{(2)} + W^{(2)^T} h^{(1)} x)$$

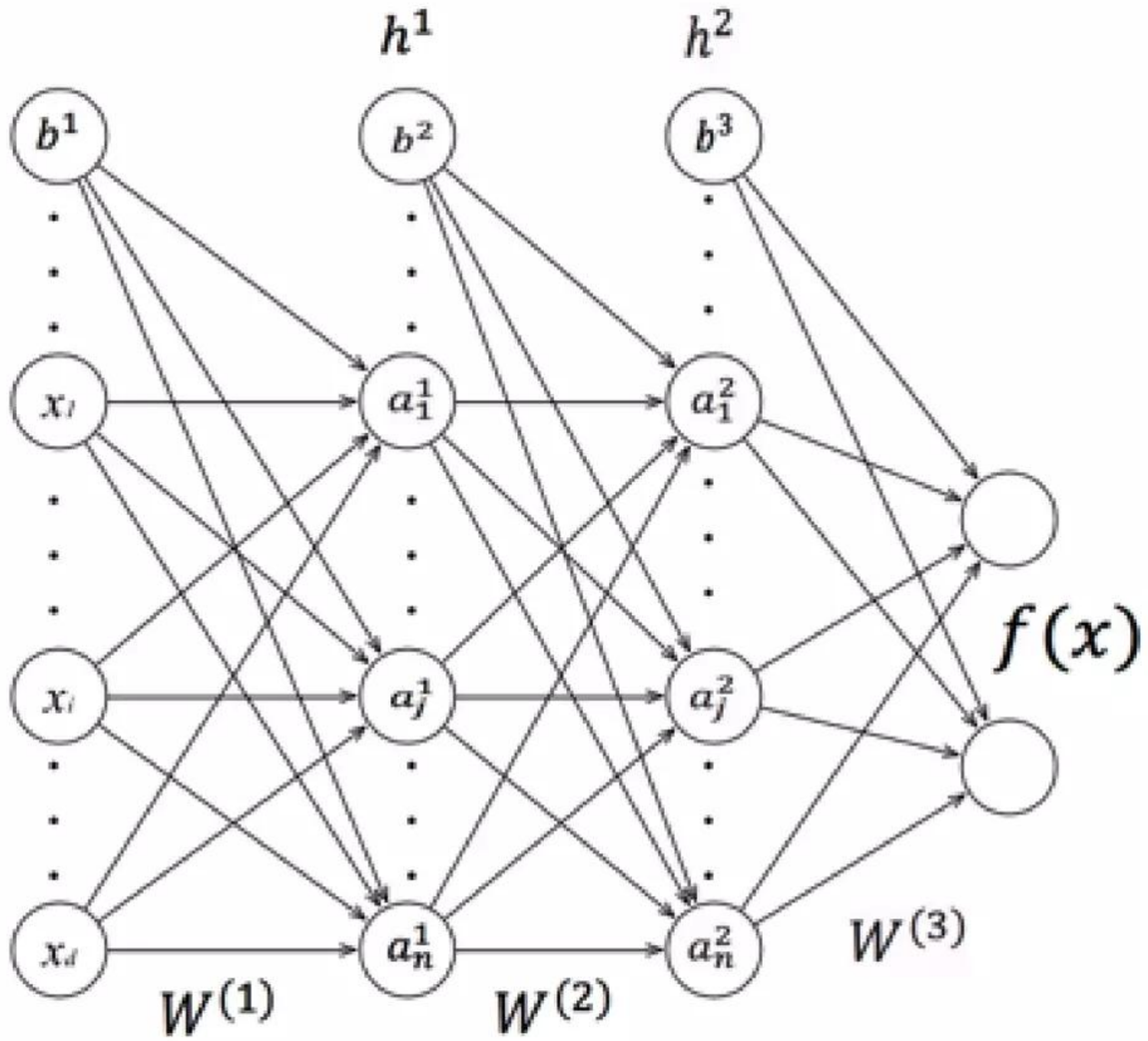


Standard neural networks

$$a^{(k)}(x) = b^{(k)} + W^{(k)} h^{(k-1)}(x)$$

$$h^{(k)}(x) = g(a^{(k)}(x))$$

$$h^{(L+1)}(x) = o\left(a^{(L+1)}(x)\right) = f(x)$$



Backpropagation

Repeat until convergence {

Initialize θ ($\theta \equiv \{W^{(1)}, b^{(1)}, \ldots, W^{(L+1)}, b^{(L+1)}\}$)

For i=1 to m iterations

Set $a^{(1)} = x^{(i)}$

Performa feed forward propagation to compute $a^{(k)}$ $for\ k = 1,2,3,\ldots,L$

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(1)}$

$\Delta_{ij}^{(k)} := \Delta_{ij}^{(k)} + a_j^{(k)} \delta_i^{(k+1)}$

}

Deep neural networks

//Pre-training

For l=1 to L

Build unsupervised training set (with $h^{(0)}(x) = x$):

$$D = \left\{h^{(l-1}(x^{(t)})\right\}_{t=1}^{T}$$

Train autoencoder on D

Use hidden layer weights and biases of greedy module to initialize the deep network parameters $W^{(l)}, b^{(l)}$

Initialize $W^{(L+1)}, b^{(L+1)}$ randomly (as usual)

//train backpropagation (supervised) with stochastic

//gradient descent

//training

Repeat until convergence {

For i=1 to m iterations

Set $a^{(1)} = x^{(i)}$

Performa feed forward propagation to compute $a^{(k)}$ $for\ k = 1,2,3,\ldots,L$

Using $y^{(i)}$, compute $\delta^{(L)}=a^{(L)} - y^{(i)}$

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(1)}$

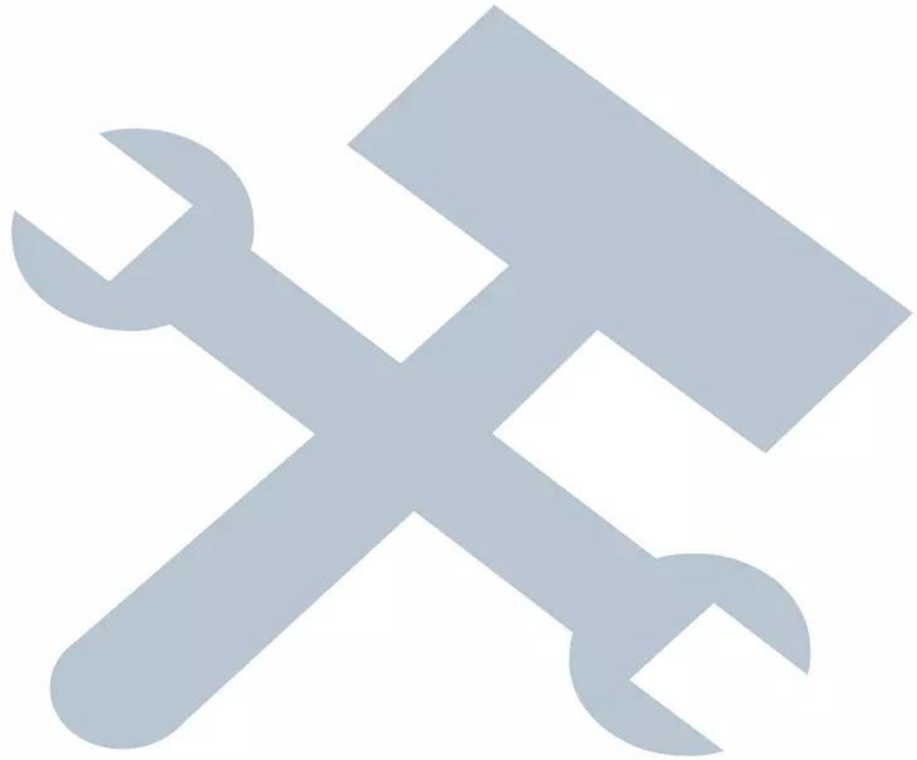
$\Delta_{ij}^{(k)} := \Delta_{ij}^{(k)} + a_j^{(k)} \delta_i^{(k+1)}$

}

[karismaet al,2016]

Pre-learning

- **Denoising autoencoder**
- Restricted boltzmann machine



Autoencoder

Decoder :

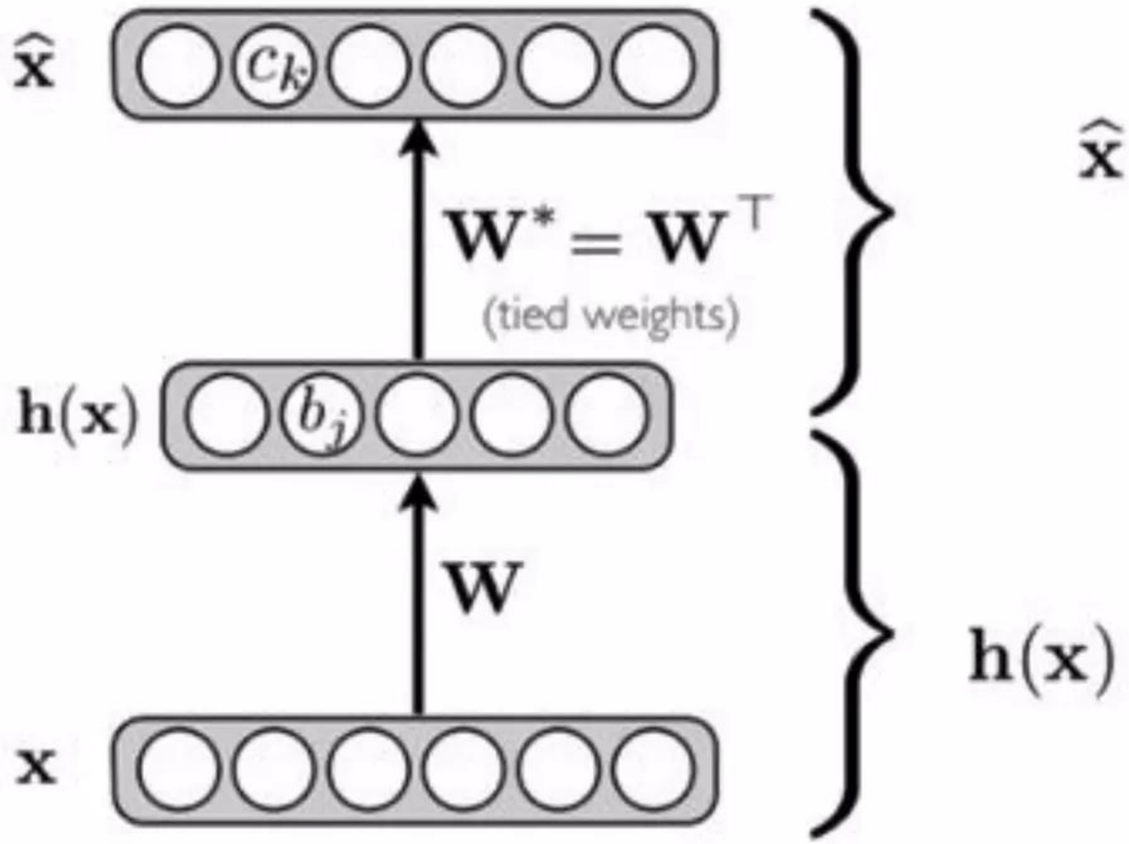
$$\hat{x} = o(\hat{a}(x))$$

$$= sigm(x + W * h(x))$$

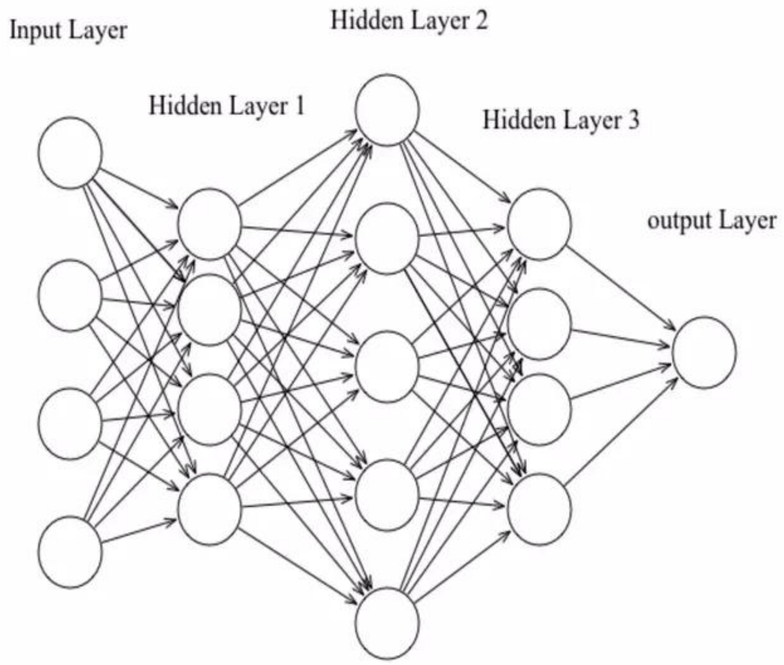
Encoder :

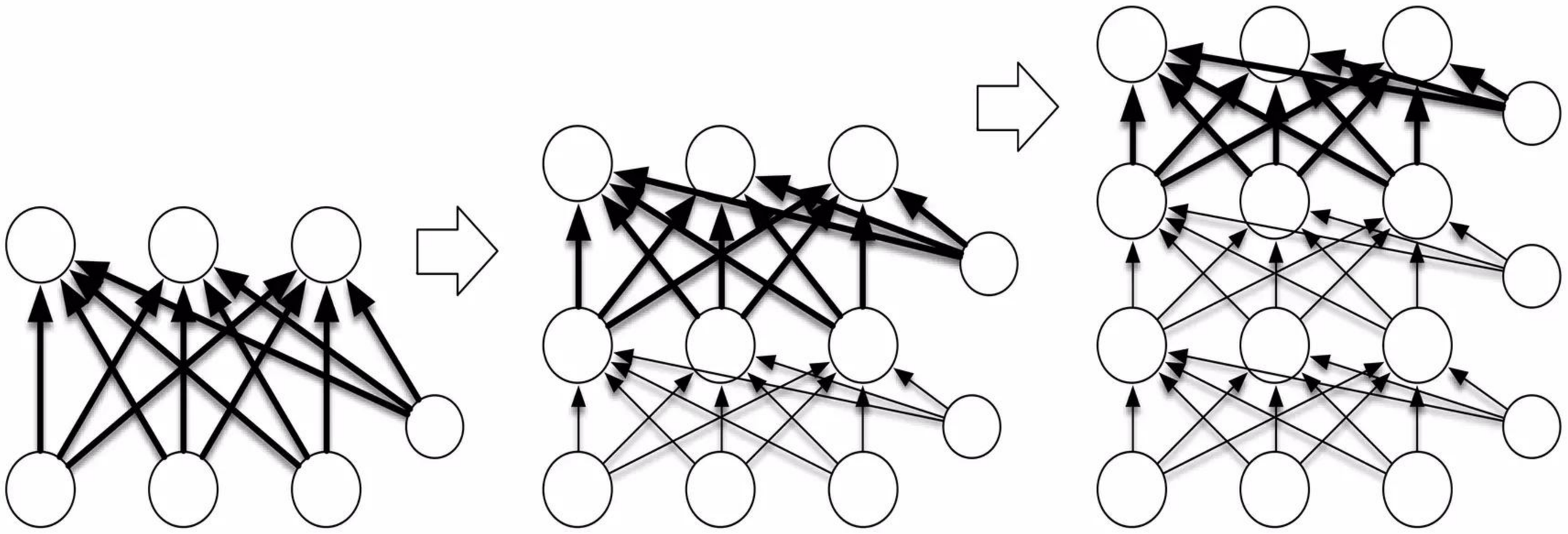
$$h(x) = g(\hat{a}(x))$$

$$= sigm(b + Wh)$$



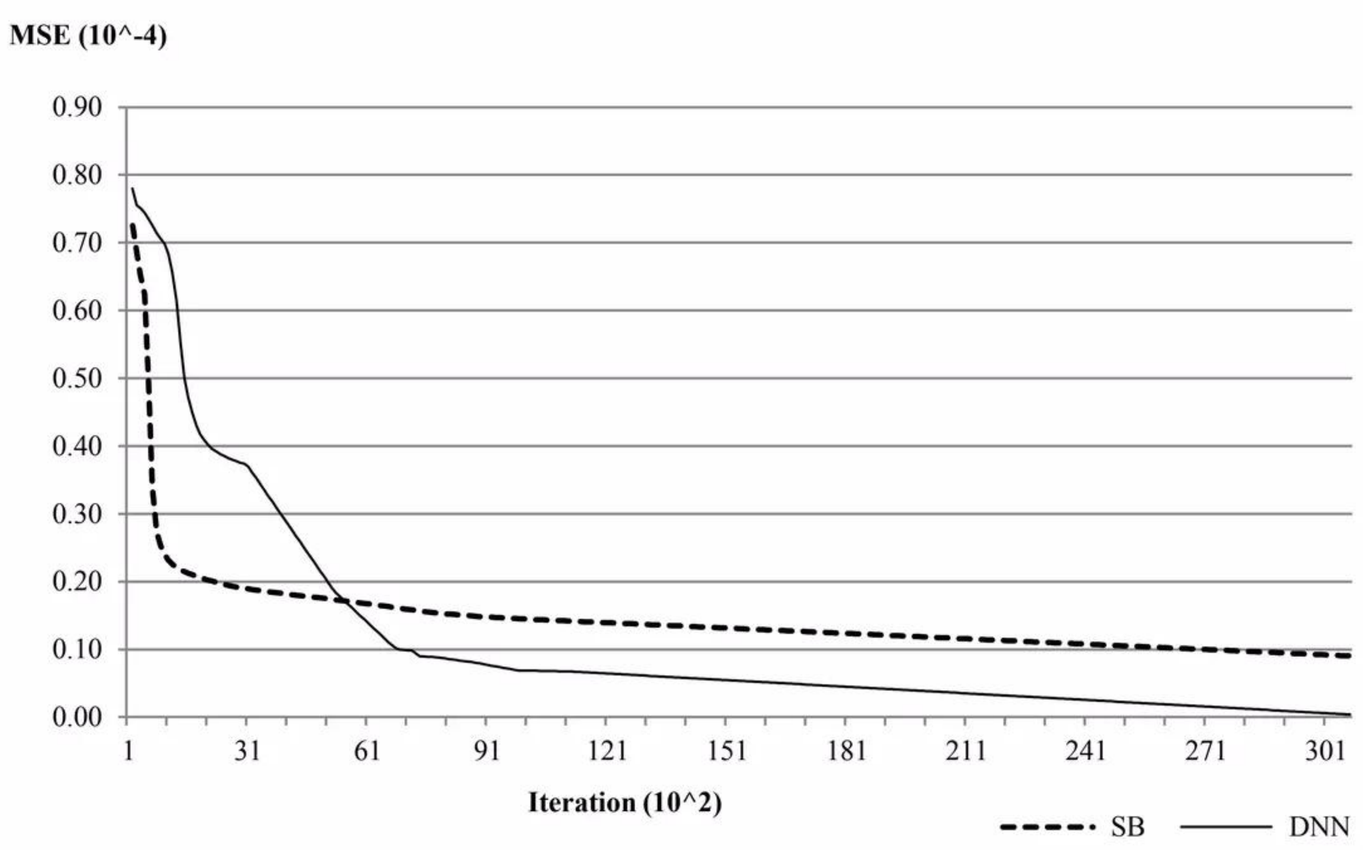
Pre-training



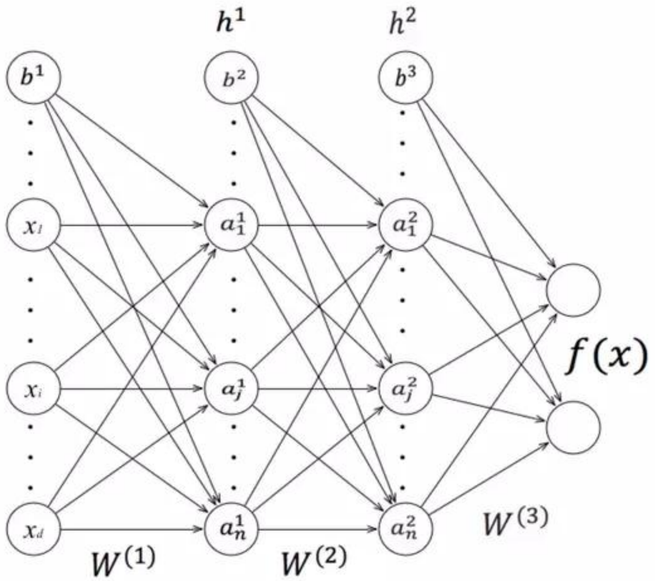


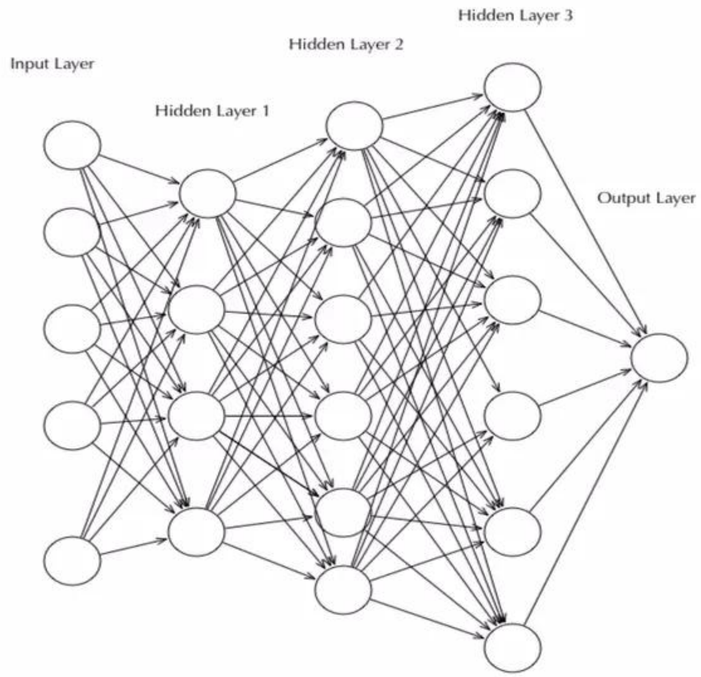
Deep neural network for bank repricing gap forcasting

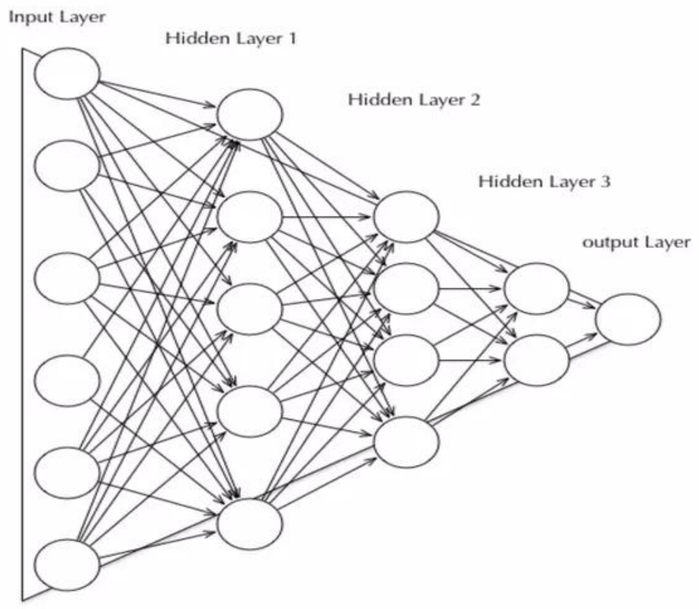
- Equals network topology
- High nonlinearity
- Almost all attributes have continuous values
- Using autoencoder
- Minimum mean squared error : *10-9*

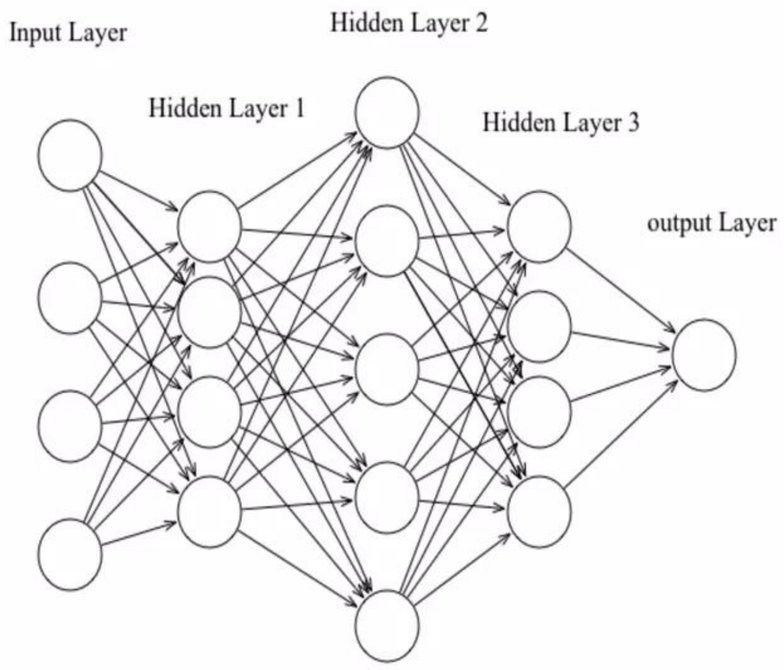


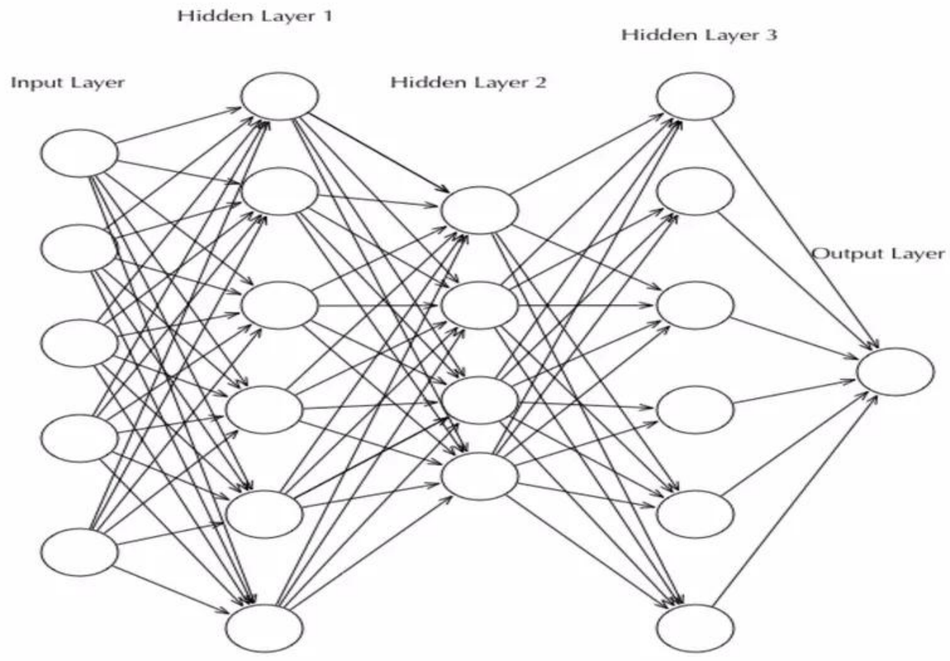
Network topologies











[karisma et al, 2016]

Algorithm Parameters

- Minimum mean squared error : *10^-8*
- Learning rate : 0.75
- Momentum : 0.5
- Topologi network : equal
- Hidden Layer : 3 Hidden Layer
- Neuron/Hidden Layer : (26, 26, 26)
- Activation function : sigmoid function
- *Autoencoder* (pre-training) parameters :
  - Minimum squared error : *10-5*
  - Max epoch: 2000
  - Learning rate : 0.5
  - Momentum : 0.75
  - Activation function : sigmoid function

Dataset

- Dataset : 4000
- Fraud : 32 (confirm fraud)
- Good transaction : 2000
- False address cases : 2157
- Suspect transaction : 500
- Attributes : +/- 102
- Non-linearity : High

Feature engineering

- Card Verification number (for BIN number)
- Postal address
- Google maps lookup (distance between shipping and billing)
- Telephone area lookup
- Credit history
- Customer order history
- Order **velocity** monitorng
- IP Geolocation
- Value Similarity (shipping and billing address, customer email and customer name)

Feature engineering (Velocity)

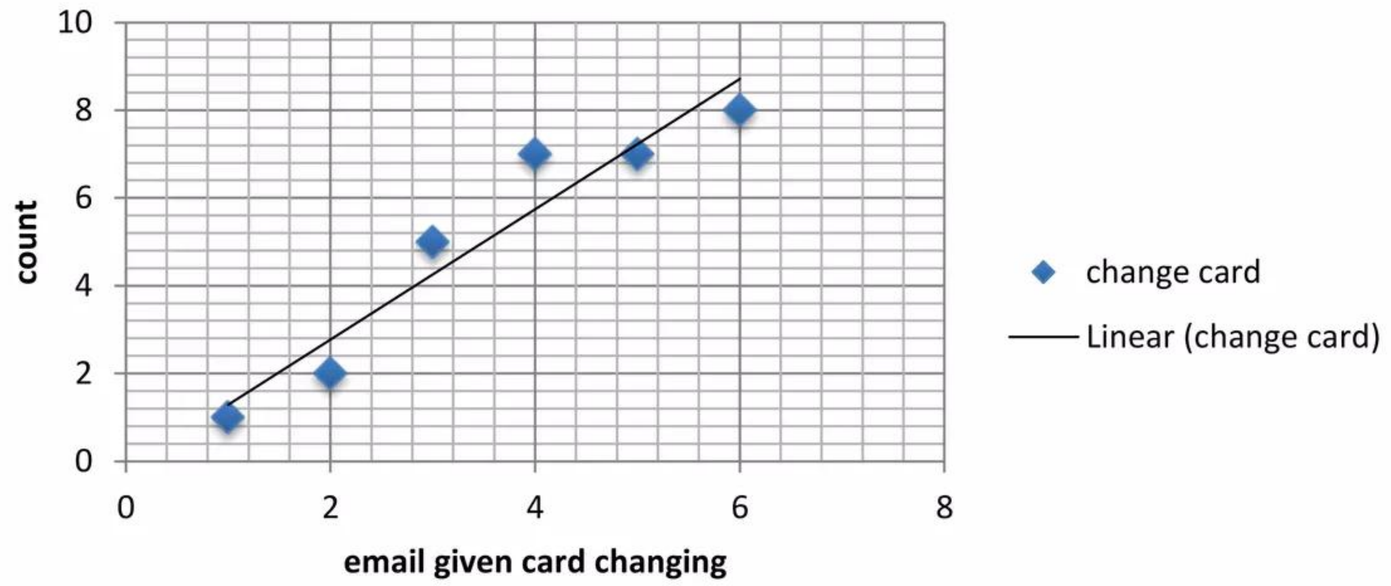
- Mask card number given : billzip, custip, email, name, shipzip. (just count)
- Mask card number given : billzip, custip, email, name, shipzip. (changing)
- Email given : billzip, custip, name, masked number, shipzip (just count
- Email given : billzip, custip, name, masked number, shipzip (changing)
- Then billzip, custip, name, and shipzip.
- so on…

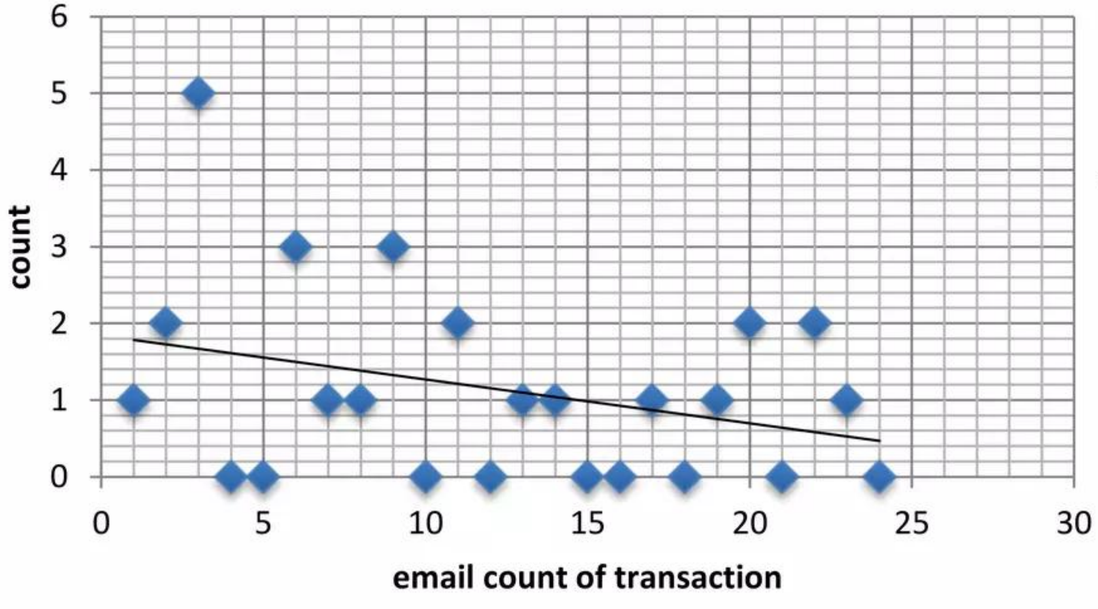
  Then compute the gradient.

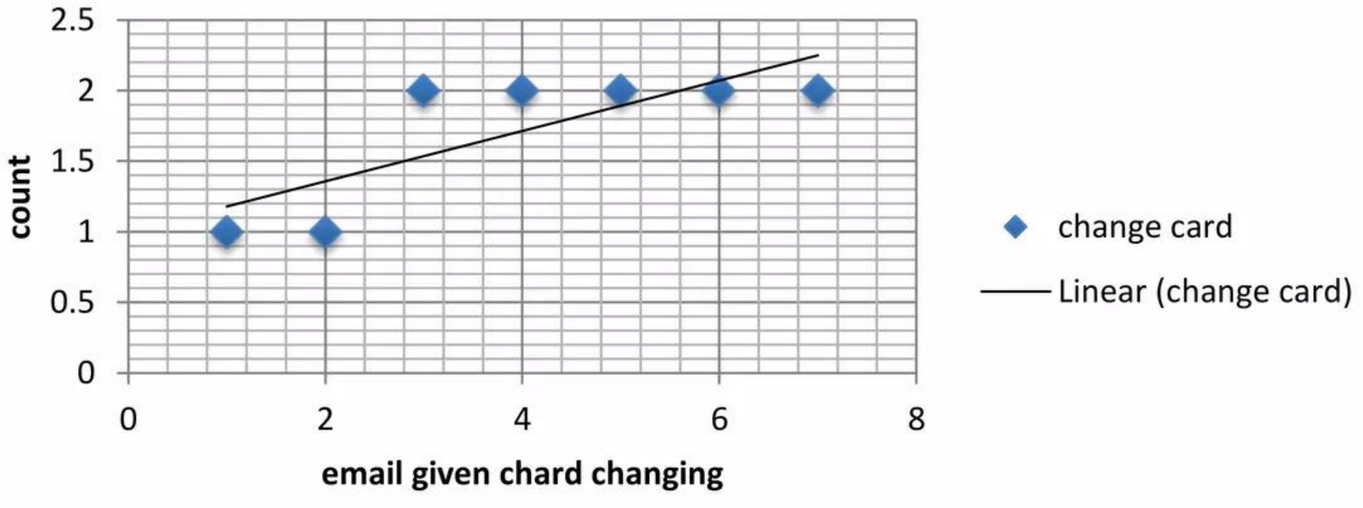
Feature engineering

It will add more than 60 features to dataset.

- Standard value
- Look-up value
- Velocity Value





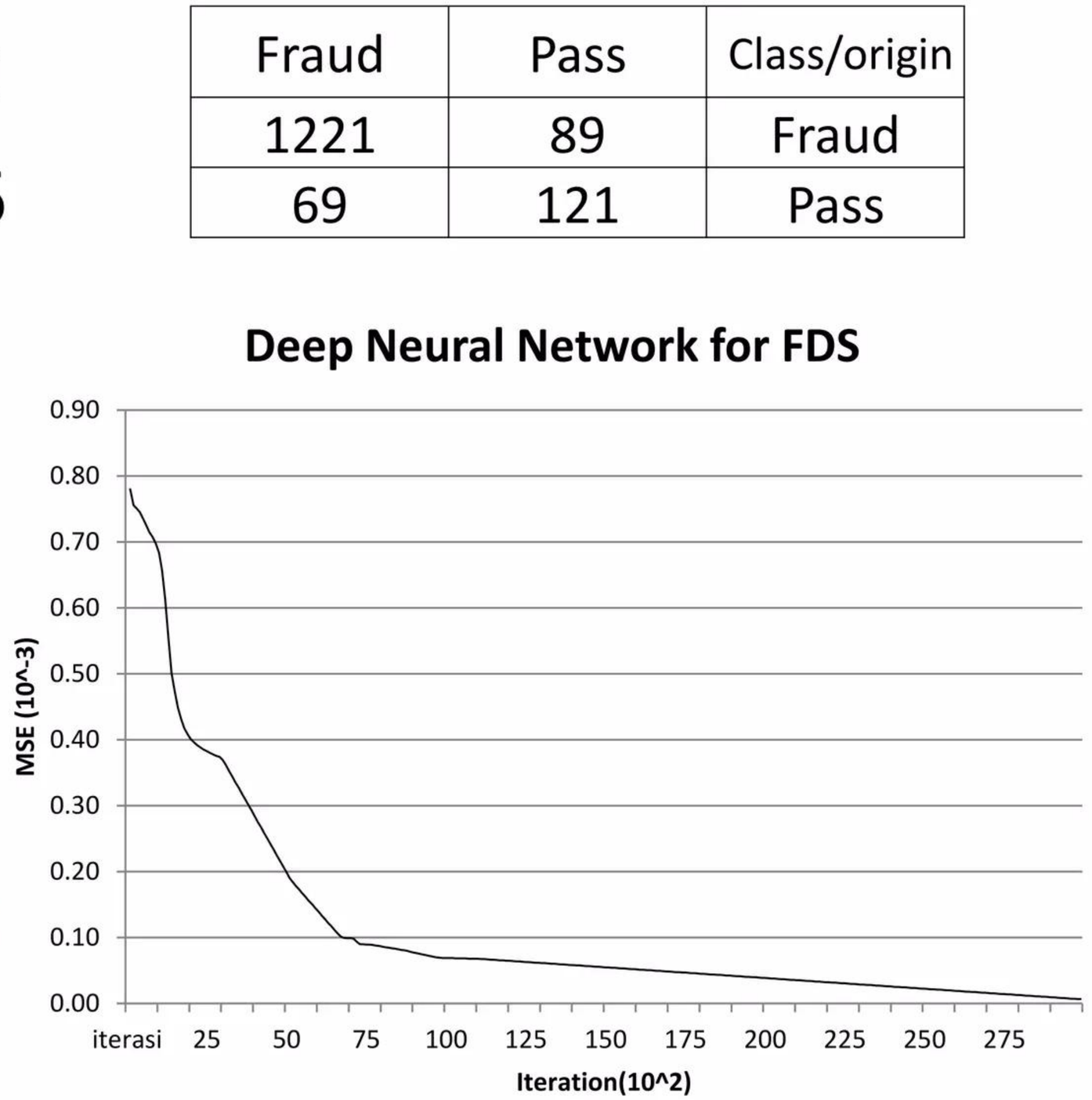


$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

Result

- Accuracy : 89.467
- Convution Matrix
- MSE : 8.31 x 10^-6

Fraud	Pass	Class/origin
1221	89	Fraud
69	121	Pass



Challenges

- Unbalancing dataset
- Fraud is transaction perspective to fraud is person perspective
- Event data (from checking page, order until transaction/checkout)
- GPU optimization
- Network model architecture
- Social network features (text and network)
- Restricted Boltzmann machine & another pre-training
- Graph theory

THANK YOU