Applied Machine Learning for Business Analytics

Lecture 5: Auto-encoders

Lecturer: Zhao Rui

Small batches bring more **noisier** gradient estimates

- Small batches can offer a regularizing effect (Wilson and Martinez, 2003), perhaps due to the noise they add to the learning process. Generalization error is often best for a batch size of 1. Training with such a small batch size might require a small learning rate to maintain stability because of the high variance in the estimate of the gradient. The total runtime can be very high as a result of the need to make more steps, both because of the reduced learning rate and because it takes more steps to observe the entire training set.

https://www.deeplearningbook.org/contents/optimization.html

Agenda

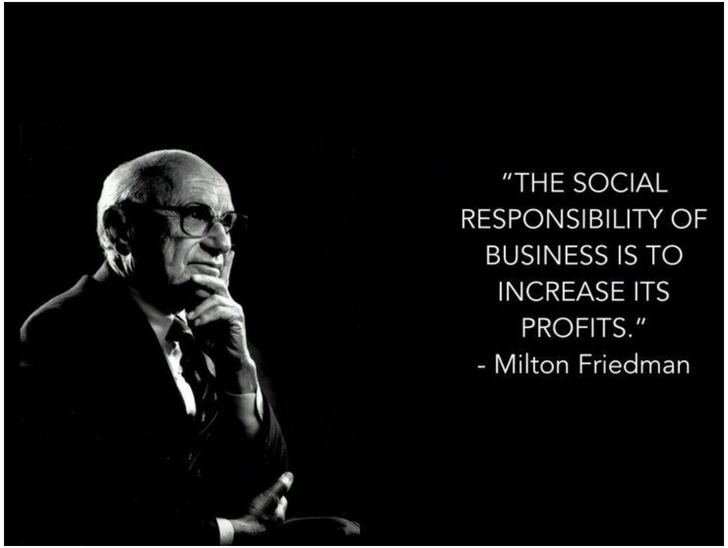
1. Project Scoping: What is one-pager?
2. Autoencoders
3. Applications of Autoencoders
4. Recommendation Systems

1. Project Scoping

Goals of ML projects

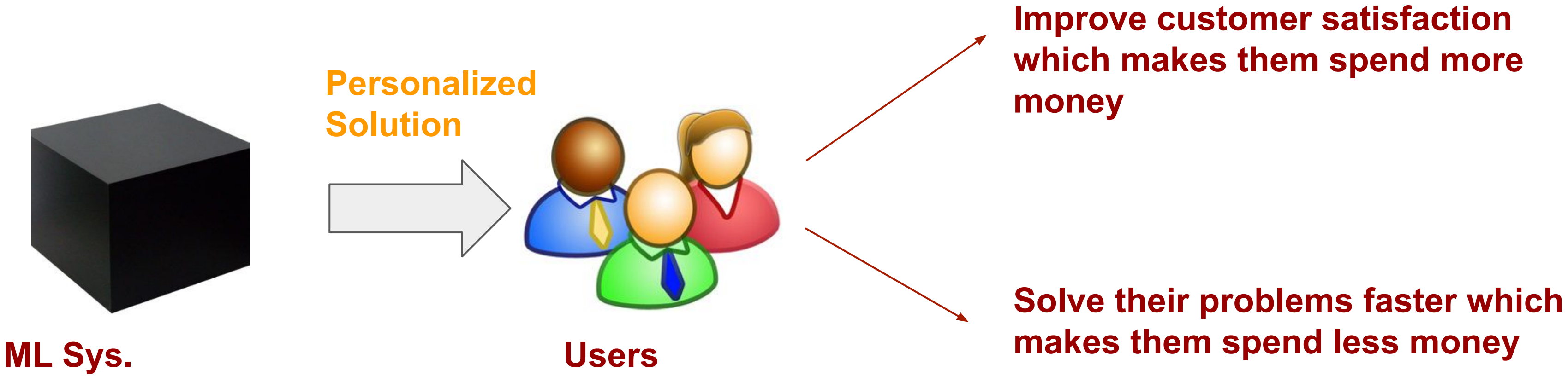
- An ML project should be aimed at increasing profits directly or indirectly.
  - Increasing sales
  - Cutting costs
  - Increasing satisfaction
  - Increasing time spent on a website

- Do we have non-profits projects? Yes
  - Climate change
  - Public health
  - Education



Connect business metrics to your machine learning models

Case study


Personalized Solution
ML Sys.
Users
Improve customer satisfaction which makes them spend more money
Solve their problems faster which makes them spend less money

Case study: movie recommendation

- When building a recommendation system for movie
    - Maximize Engagement
    - Maximize Revenue from sponsored content
        - Click more, ads fee more
    - Minimize the spread of restricted content

How to set goals?

- Goals: General Purpose of a Project
  - Maximize users' engagement while minimizing the spread of violent content and maximize revenue from sponsored content
- Objectives: Specific steps on how to achieve the above goals
  - Filter out unclasificated movies
  - Rank movies by quality
  - Rank movies by their ads fee
  - Rank movies by engagement: how likely users will watch it

How to combine these two targets via ML systems?

Multi-objective system

- Rank Movies by quality
  - Predict films’ rating
  - Minimize Rating_loss: loss between predicted rating and true rating

- Rank movies by engagement: how likely users will watch it
  - Predict watch times
  - Minimize Engagement_loss: loss between predicted watch times and true times

Solution: combine different models

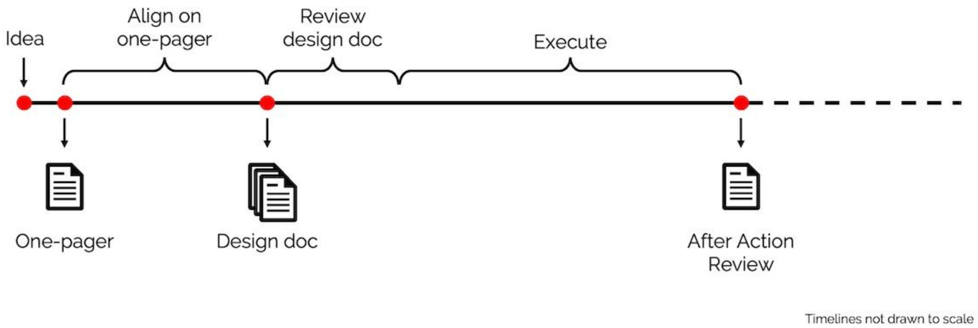
- Train two models
  - Model A: rating_loss
  - Model B: engagement_loss
  - Rank movies by \alpha*pred_modelA + \beta*pred_modelB

Decouple different objectives

- Easier for training
- Easier to tweak our systems
  - No need to retrain the whole system if weights for different objectives are changed
- Easier for maintenance
  - Different objectives might need different maintenance schedules

One-pager for machine learning projects

- Amazon Writing Style Tip
  - https://medium.com/fact-of-the-day-1/amazon-writing-style-tip-a349b4bd3839
- How to write design documents for data science/machine learning projects?
  - https://eugeneyan.com/writing/writing-docs-why-what-how/



Three types of documents required during projects

How to use the framework to structure your docs

Here are some examples of using Why-What-How to structure a one-pager, design doc, after-action review, and my writing on this site.

	Why?	What?	How?
One-Pager	• Problem or opportunity • Hypothesized benefits	• Success metrics • Constraints	• Deliverables • Define out-of-scope
Design Doc	• Why the problem is important • Expected ROI	• Business / product requirements • Technical requirements & constraints	• Methodology & system design • Diagrams, experiment results, tech choices, integration
After-action Review	• Context of incident • Root cause analysis (5 Whys)	• Tangible & intangible impact • Estimates (e.g., downtime, $)	• Follow-up actions & owners
Writing on this site	• Why reading the post is important (e.g., anecdotes)	• The topic being discussed (e.g., documents we write at work)	• The insight being shared (e.g., Why-What-How, examples)

One-pager example

Why: Our data science team (in an e-commerce company) is challenged to help customers discover products easier. Senior leaders hypothesize that better product discovery will improve customer engagement and business outcomes.

What: First-order metrics are engagement (e.g., CTR) and revenue (e.g., conversion, revenue per session). Second-order metrics include app usage (e.g., daily active users) and retention (e.g., monthly active users). Constraints are set via a budget and timeline.

How: The team considered several online (e.g., search, recommendations) and offline (e.g., targeted emails, push notifications) approaches. Their analysis showed the majority of customer activity occurs on product pages. Thus, an item-to-item (i2i) recommender—on product pages—is hypothesized to yield the greatest ROI.

Appendix: Breakdown of inbound channels and site activity, overview of the various approaches, detailed explanation on recommendation systems.

Achieve alignment

Make alignment with business/product owners in the following terms:

- Business Problem
  - Our platform has so many voucher hunters
- Hypothesized Benefits
  - Effective fraud detection model will save cost
- Success metrics
  - First-order metrics are customer acquisition cost (voucher campaign)
  - Second-order metrics are users retention rate.
- Constraints
  - Low False Positive Rate
- Deliverables
  - ML Fraud detection system

WHY

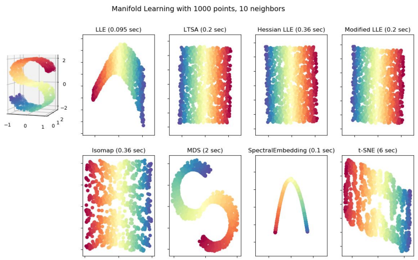
WHAT

HOW

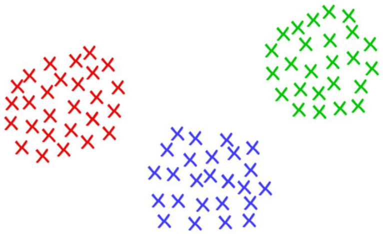
2. Autoencoders

Unsupervised learning

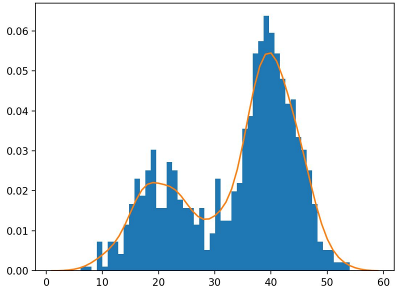
- Given the data x without labels
- Goal: Learn hidden structure (low dimension)



Representation Learning
Data lies on a low-dimensional manifold



Clustering
Group data points based their similarity



Density Estimation
Estimate data probability p(x) from data x1, x2, ...,, xn

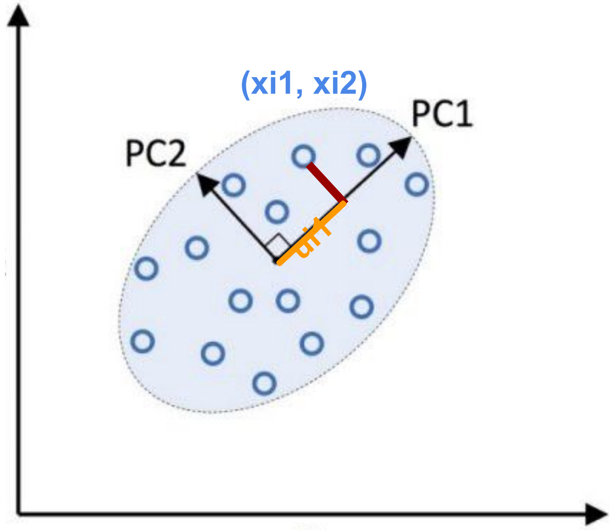
Principal component analysis: maximize variance

- PCA aims to find the directions of maximum variance in high-dimensional data and projects it onto a new subspace with equal of fewer dimensions than the original one
- Goal: Learn hidden structure (low dimension)

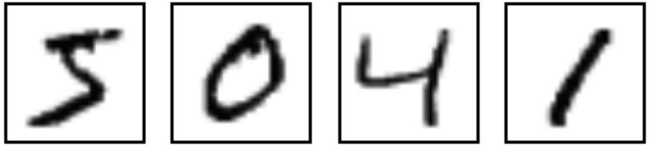
Original Space | Projection Matrix | New/Latent Space

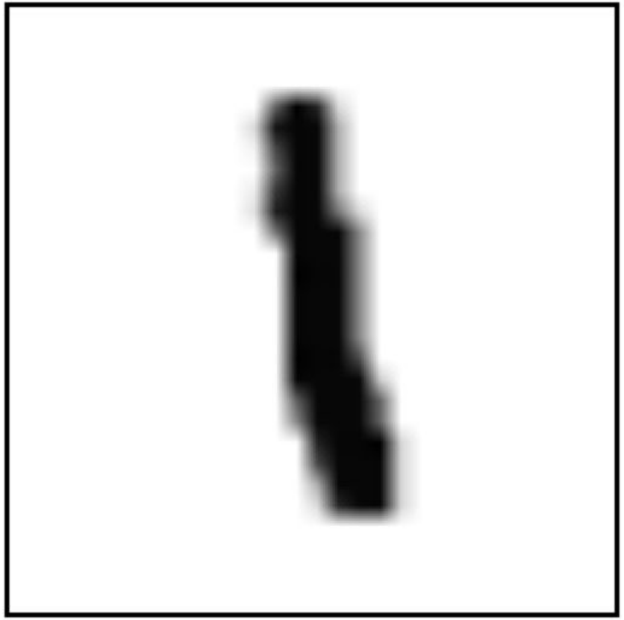
$$\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{n1} & x_{n2} \end{bmatrix} \times \begin{bmatrix} w_{11} \\ w_{21} \end{bmatrix} = \begin{bmatrix} u_{11} \\ u_{21} \\ \vdots \\ u_{n1} \end{bmatrix}$$

PC1



MNIST dataset

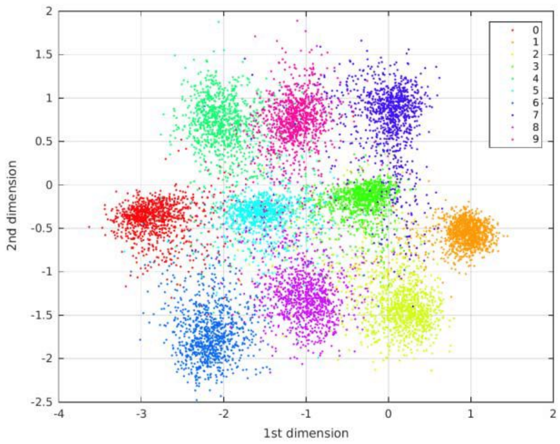




$$\approx \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&.6&.8&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.5&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.7&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&.9&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&.3&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$

PCA for MNIST visualization

- Each image has 28 by 28 pixels -> 28 by 28 matrix -> 784 dimensional vector
- Using PCA, find a project matrix $\mathbf{W} \in R^{784 \times 2}$
- After project, each image can be encoded into a 2-dimensional space



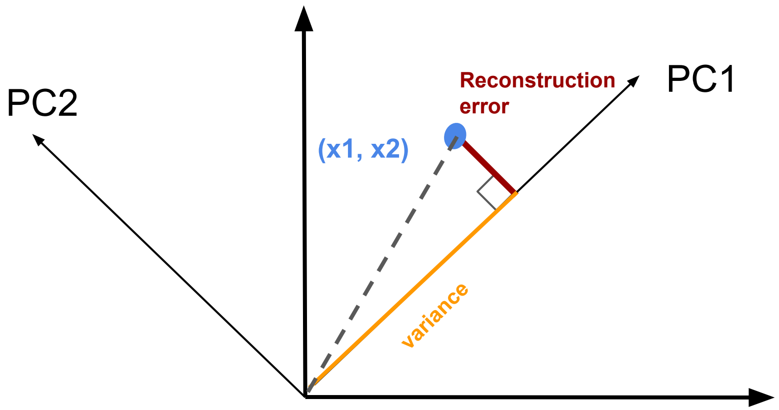
PCA: minimize reconstruction error

- PCA aims to find a linear subspace that minimize the distance of the project in a least-square sense

$$\underset{\mathbf{W}}{\text{minimize}} \quad ||\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^T||_F^2$$

$$\text{subject to} \quad \mathbf{W}^T\mathbf{W} = I$$

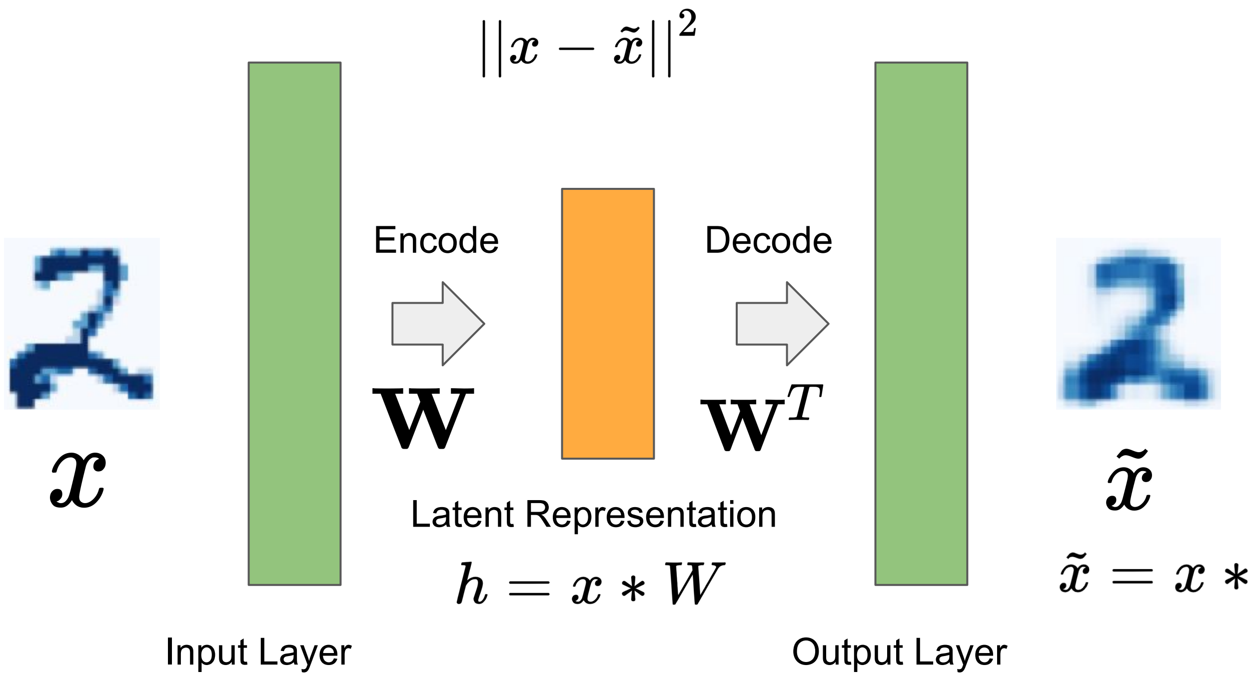
W's shape is (d, h) and h < d



Reconstruction Error + **Variance** = **Constant**

minimize *maximiz*

PCA in neural network format



$$\tilde{x} = x * W * W^T$$

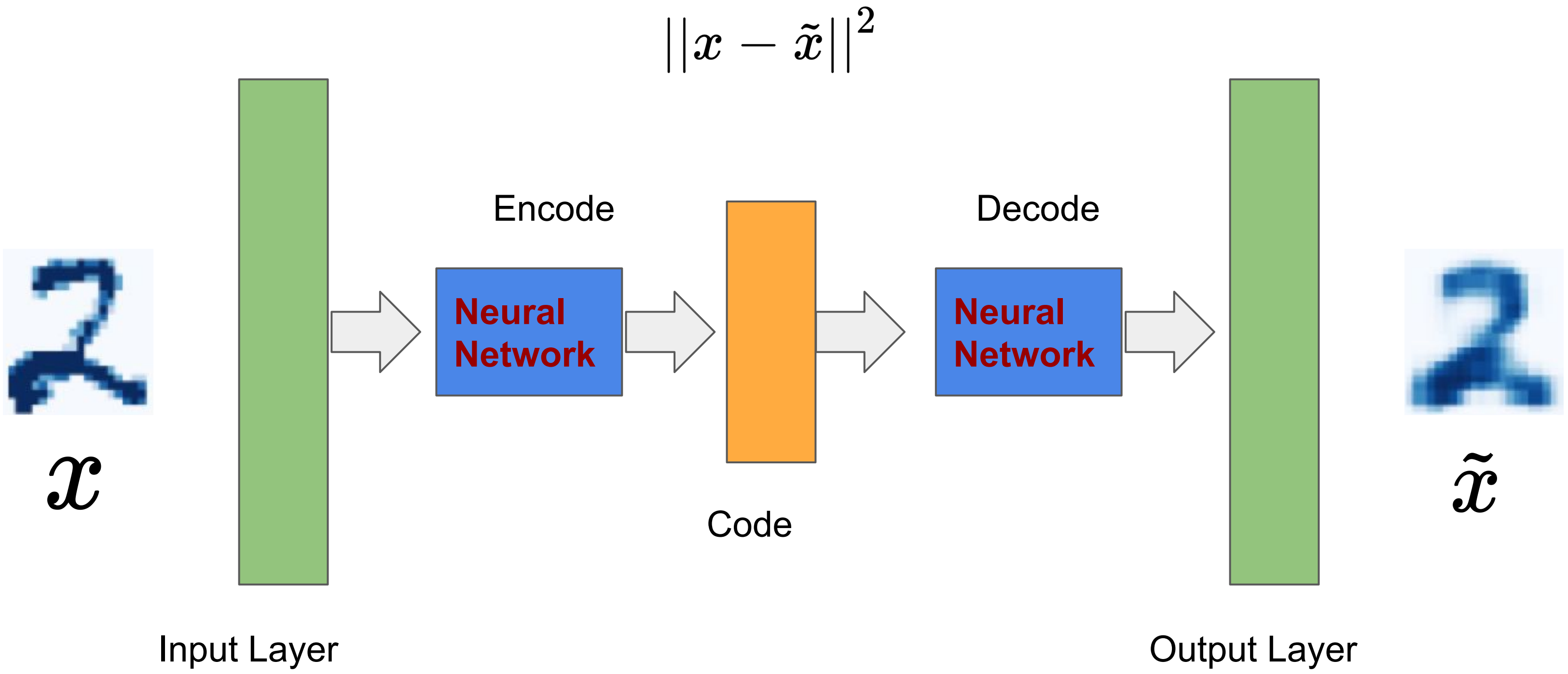
- Linear
- Orthogonal Columns in W

- Output of hidden layer is the latent representation

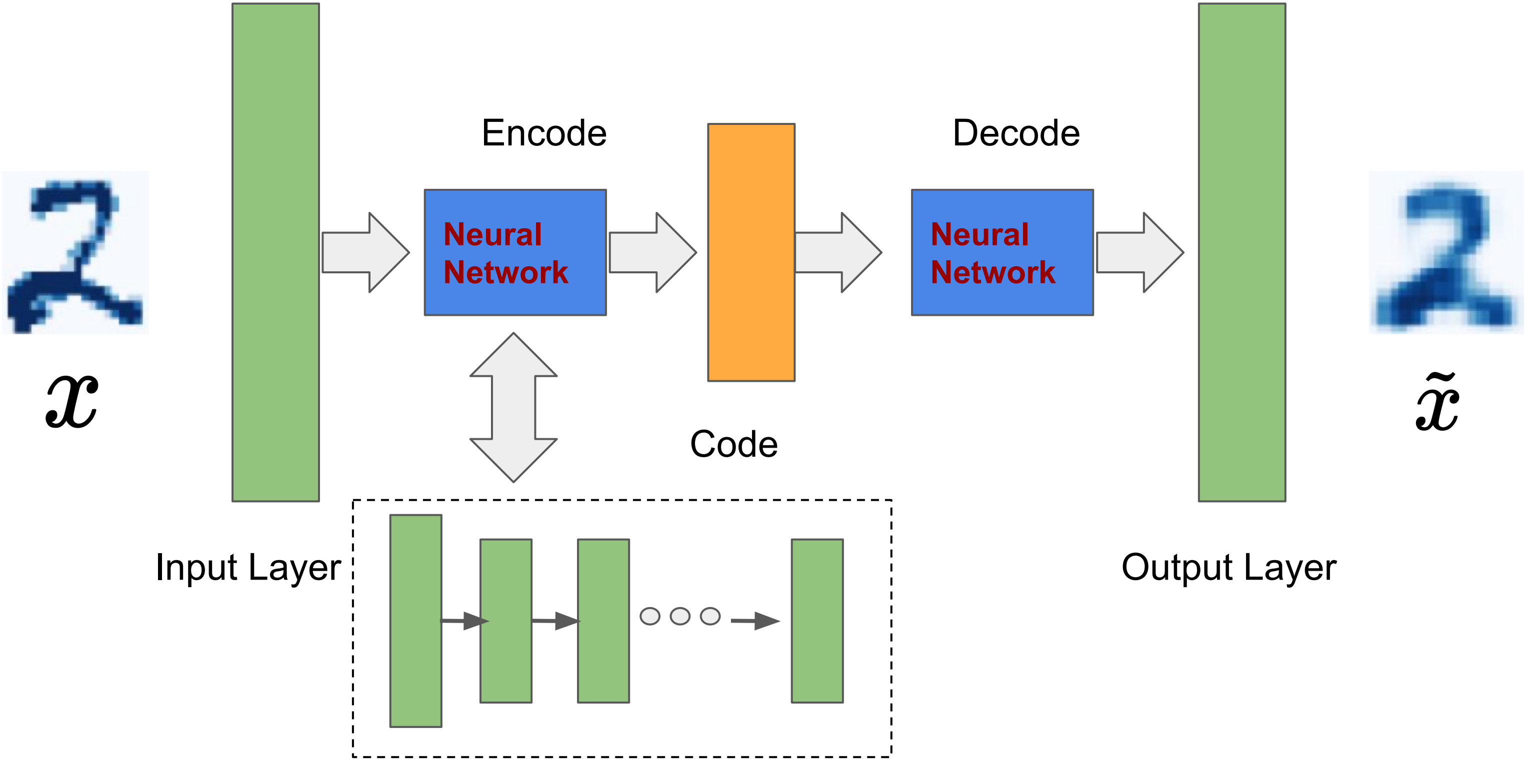
More Details on PCA: http://www.cs.cornell.edu/courses/cs4786/2016sp/lectures/lec03.pdf

- Non-linear relationship between original representation and latent features
- Which machine learning models is used for **nonlinear approximation**?

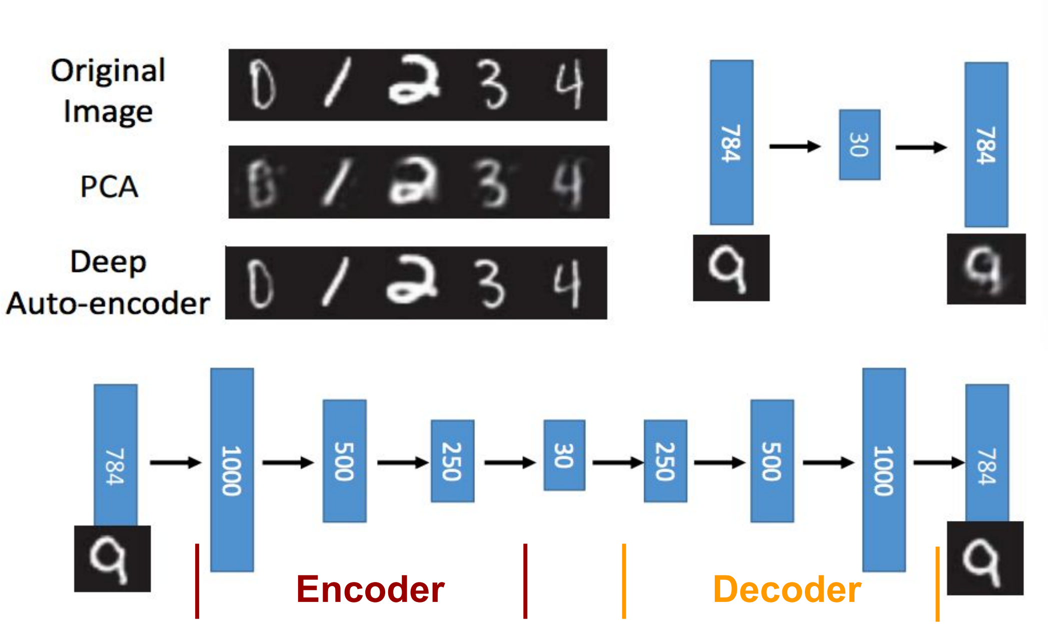
Autoencoder: nonLinear



Deep autoencoder

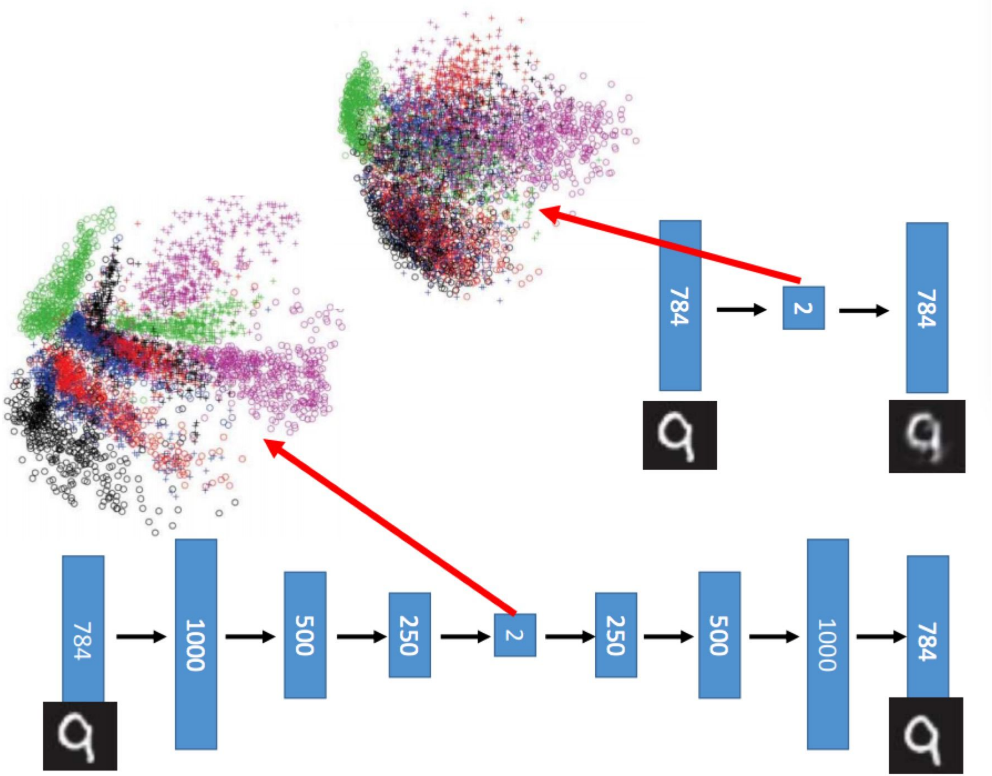


Deep Autoencoder vs PCA

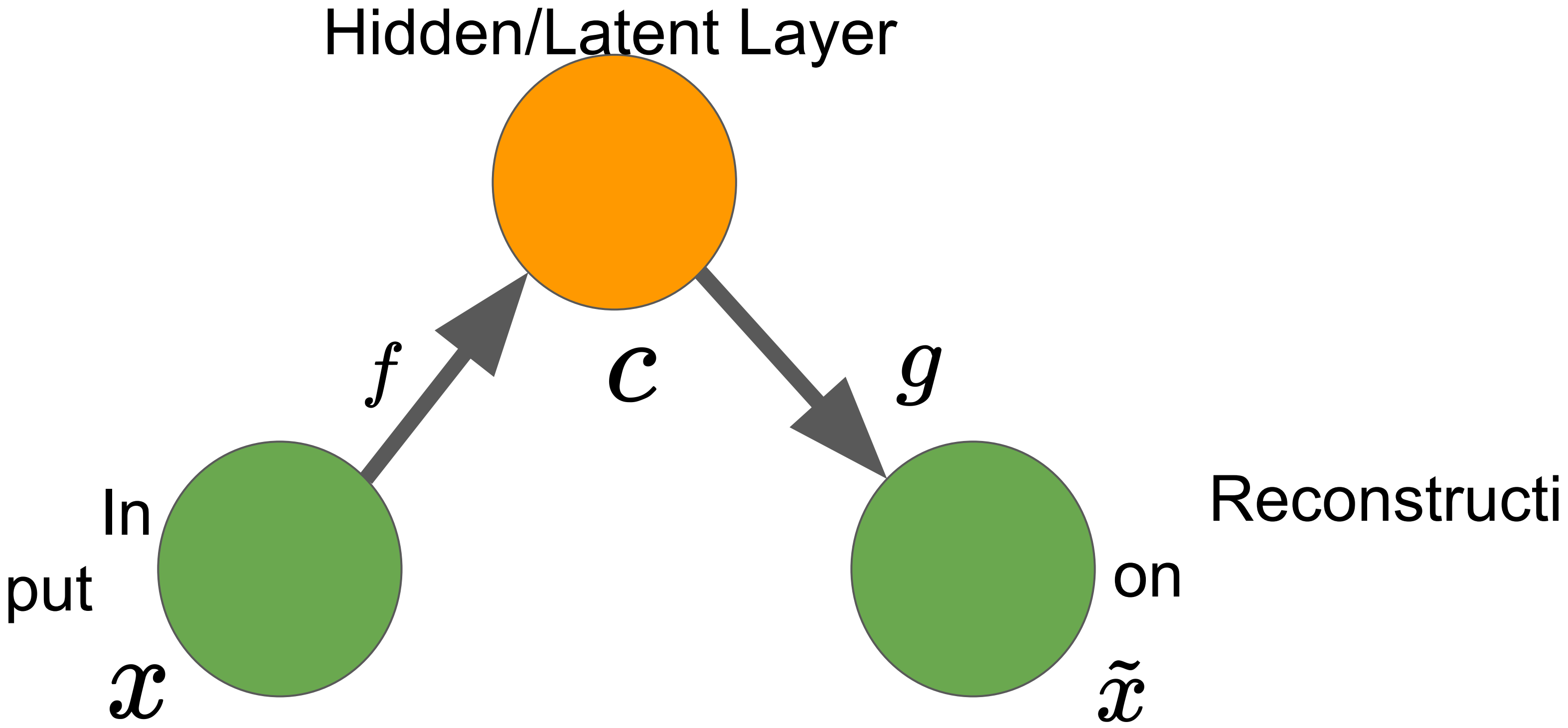


Symmetric Structure

Deep Autoencoder vs PCA



Structure of autoencoder

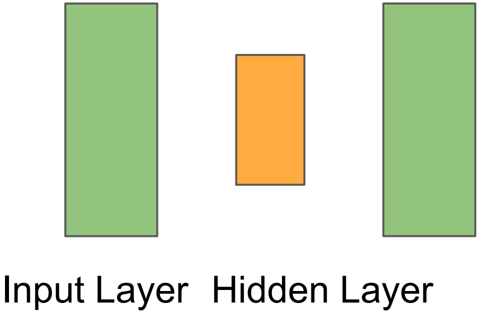


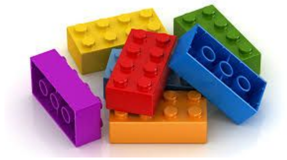
Undercomplete autoencoder

- Simply copy input to output without learning anything useful
  - The autoencoder just mimic the identify function
  - Reconstruct the training data perfectly
  - Overfitting
- To avoid the above issues, we should use undercomplete autoencoders
  - The hidden layer size c is small compared to the original feature dimensionality

Sandwich architecture in autoencoder

- Forcing c (hidden layer size) is less than d (the input layer size)
  - Learn the important features
  - Information bottleneck:
    - A kind of trade-off between compression and retaining information



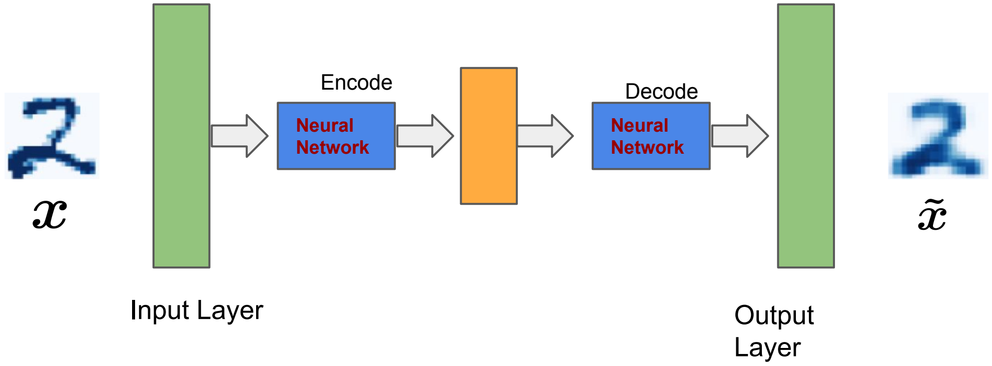


Original **6** Bricks

Can we use only **4** bricks to rebuild the previous shape?

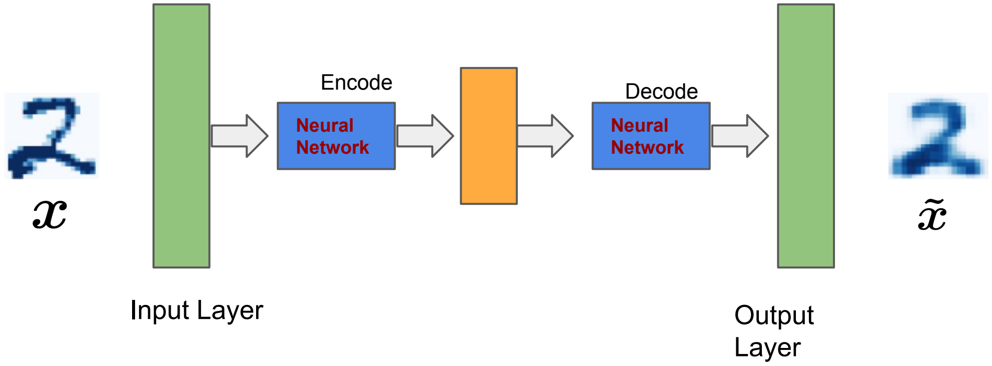
Optimization targets

- For Autoencoder, the training objective is to minimize $||x - \tilde{x}||^2$
- Hidden representation is what we really want to learn



Unsupervised or Self-supervised

- Autoencoder is one kind of self-supervised learning $||x - \tilde{x}||^2$
- Input is x, target is x
- Pretend there is part of the input you do not know and predict that



Build autoencoders in Keras

https://blog.keras.io/building-autoencoders-in-keras.html

Regularized autoencoder

Add constraints in case the identity transformation is learned, i.e., overfitting

Sparse autoencoders

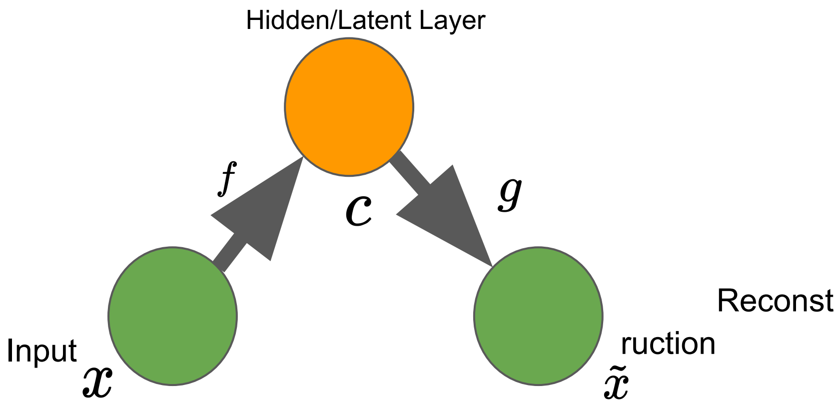
- Constrain on c that penalizes it from dense
- Regularization on output of encoder, not parameters

$$L(x, g(f(x))) + \Omega(c)$$

- `kernel_regularizer` : instance of `keras.regularizers.Regularizer`
- `bias_regularizer` : instance of `keras.regularizers.Regularizer`
- `activity_regularizer` : instance of `keras.regularizers.Regularizer`

Example

```
from keras import regularizers
model.add(Dense(64, input_dim=64,
                kernel_regularizer=regularizers.l2(0.01),
                activity_regularizer=regularizers.l1(0.01)))
```

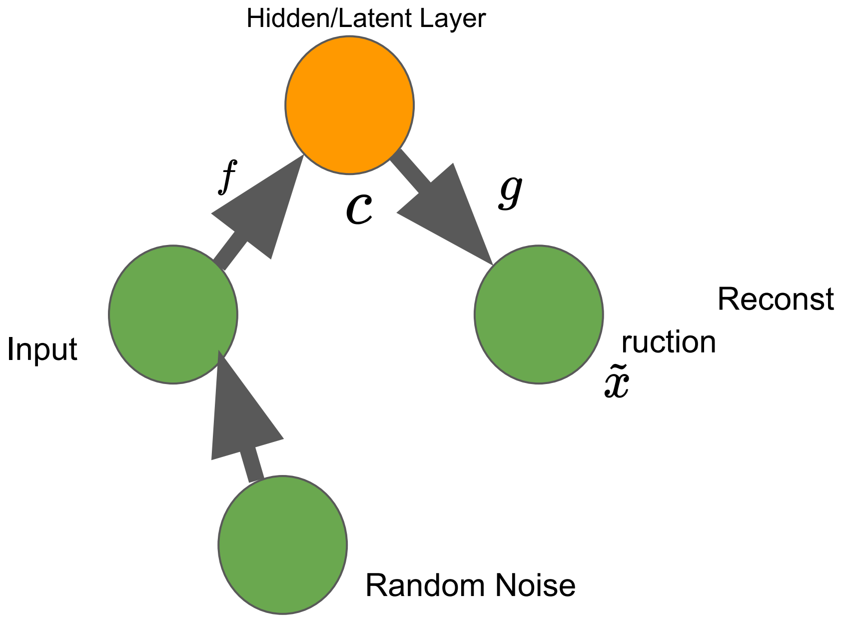


Denoising autoencoders

- Add noise into original data points
- Still reconstruct the original data points

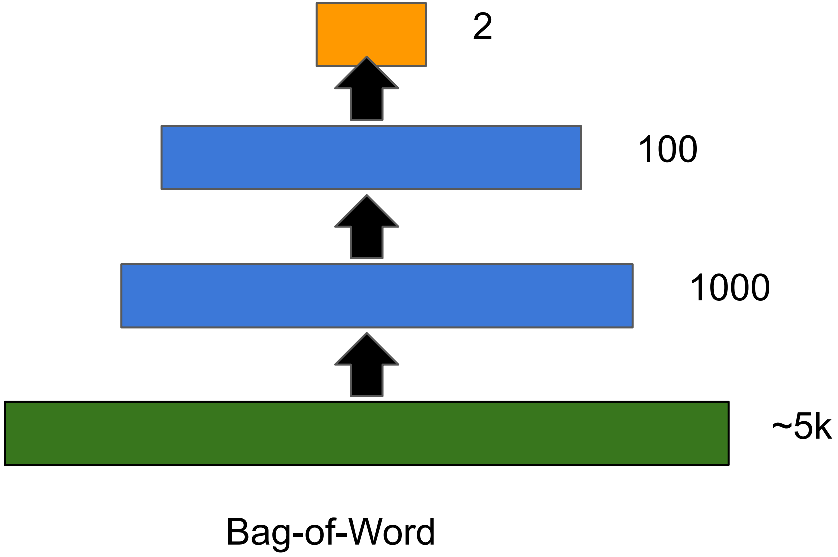
$$L(x, g(f(\bar{x})))$$

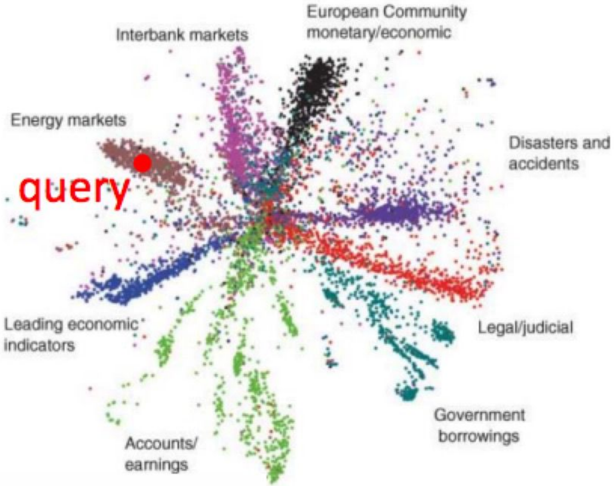
Corrupted copy of x



3. Applications of Autoencoders

Better representation

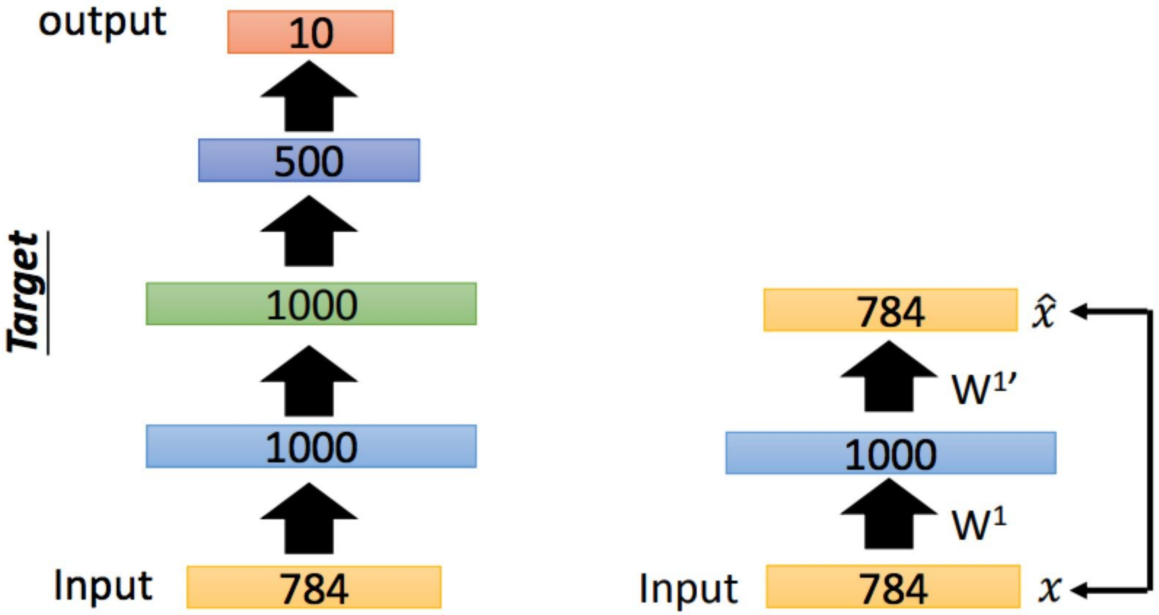




LSA: project documents to
2 latent topics

Pre-training deep neural network

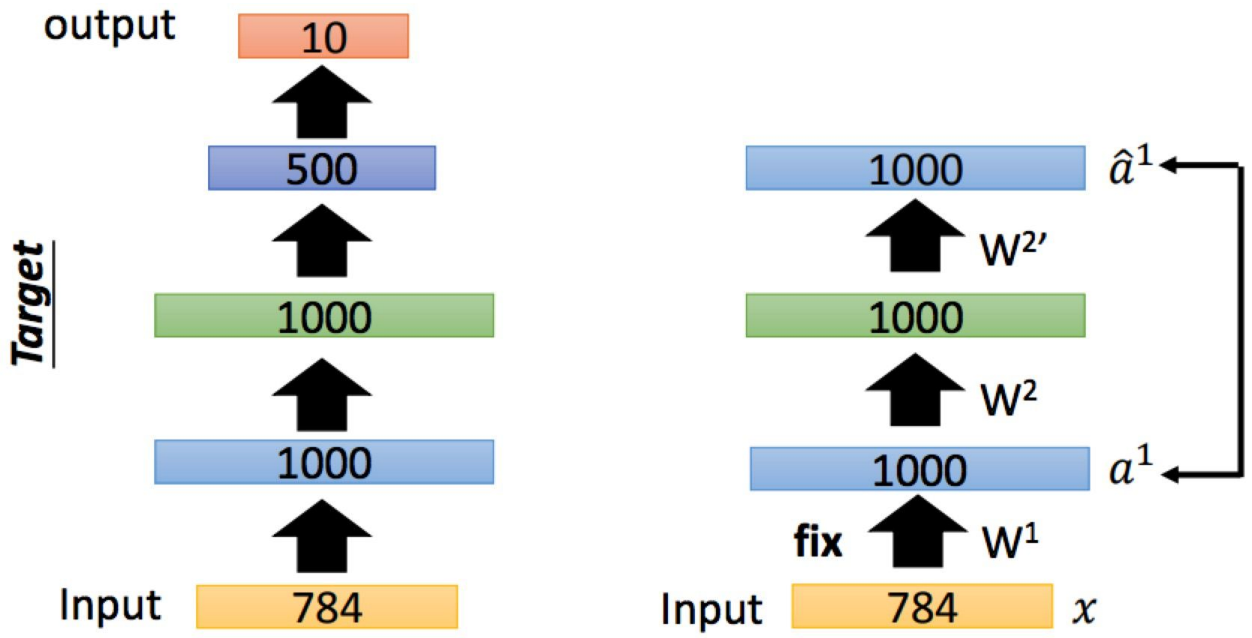
- Greedy Layer-wise Pre-training for W1



Unsupervised

Pre-training deep neural network

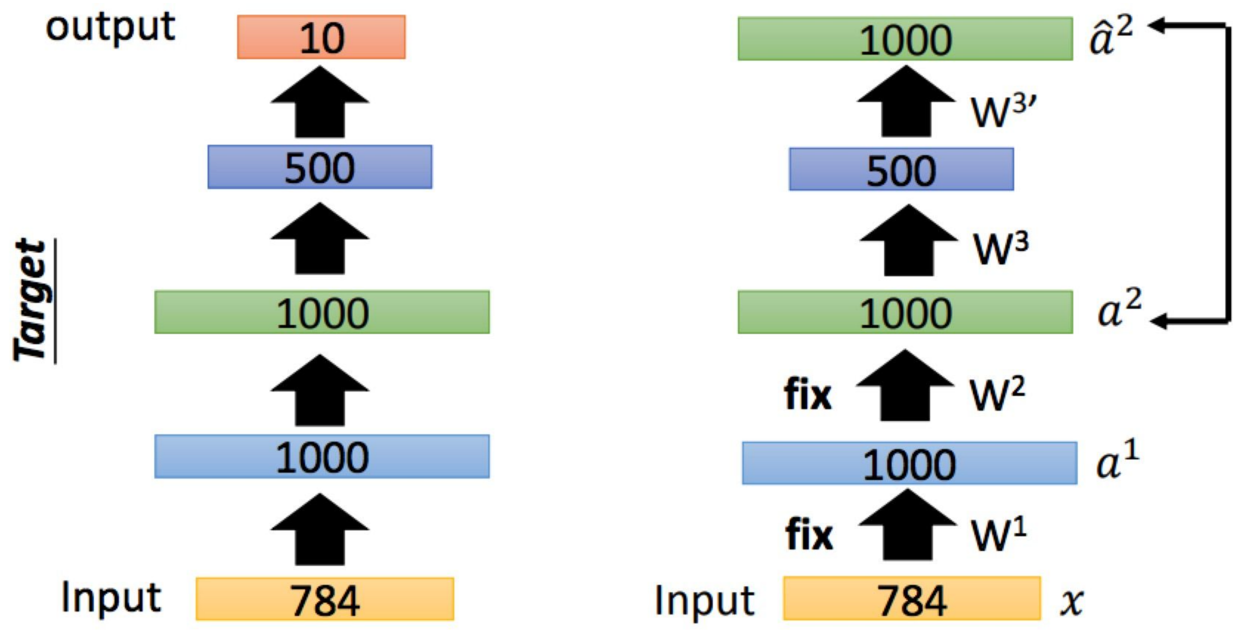
- Greedy Layer-wise Pre-training for W2



Unsupervised

Pre-training deep neural network

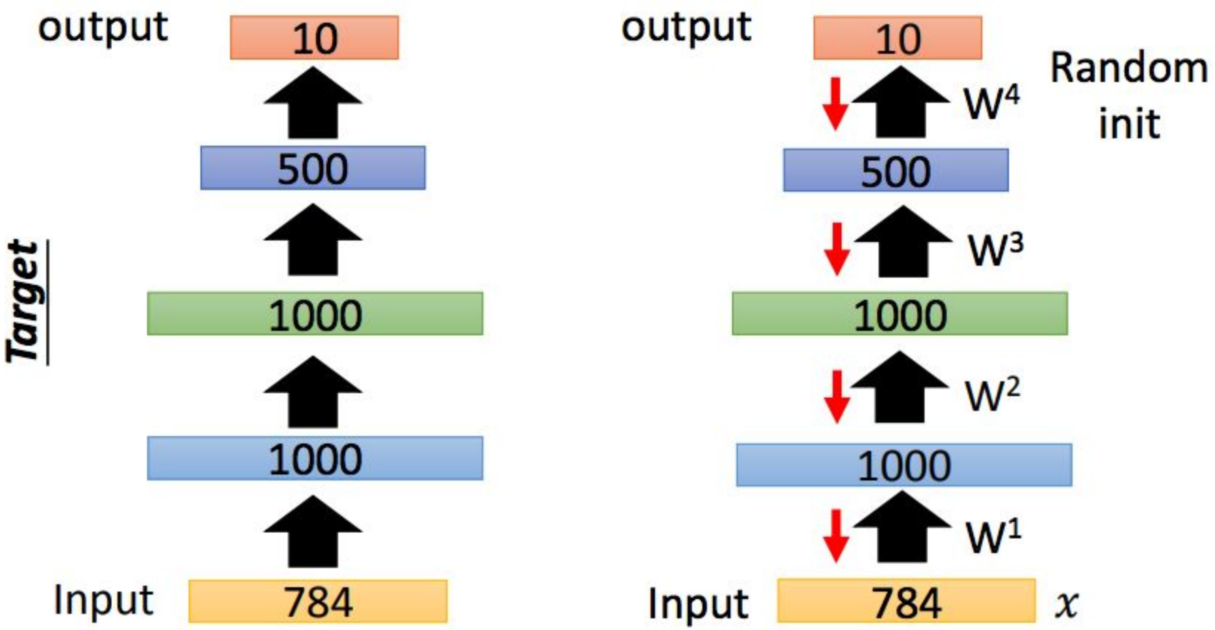
- Greedy Layer-wise Pre-training for W3



Unsupervised

Pre-training deep neural network

- Fine-tune by backpropagation



supervised

4. Recommendation Systems

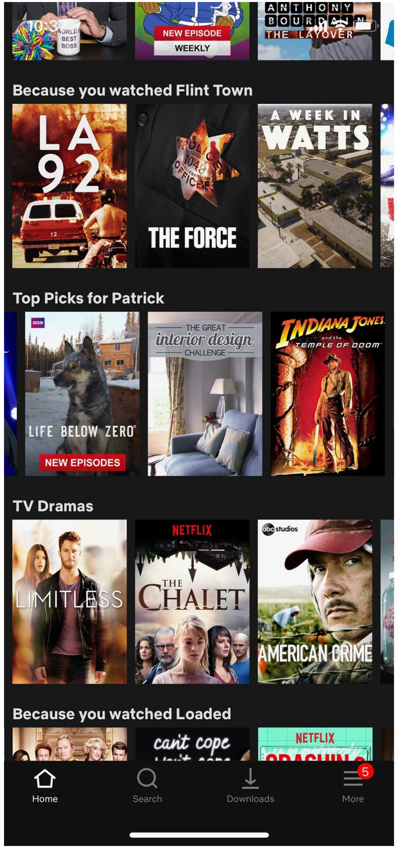

Arjun Narayan
@narayanarjun
Follow

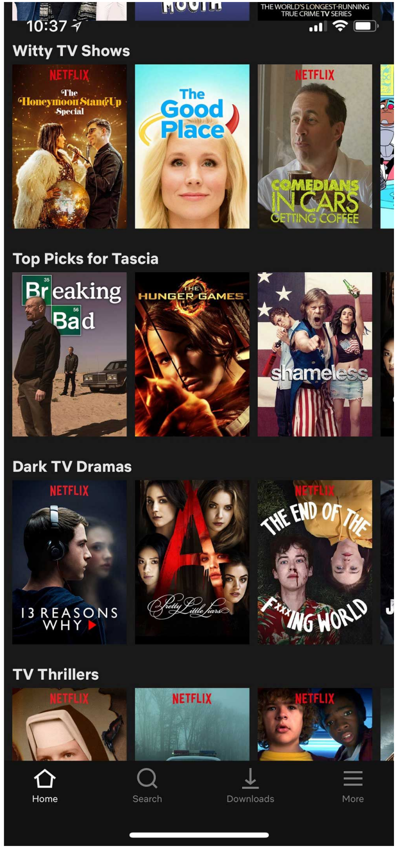
The two best performing public stocks of the decade - Netflix (+3700%) and Domino's Pizza (+3000%) - perfectly epitomize the 2010s. You either build the world's most advanced machine learning content recommender system, or make a better pizza sauce, there's no middle ground.

1:20 PM - 27 Dec 2019

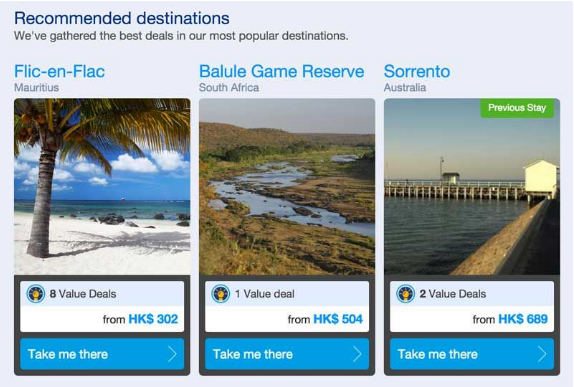
3,926 Retweets 20,086 Likes

183 3.9K 20K


Because you watched Flint Town
Top Picks for Patrick
TV Dramas
Because you watched Loaded
Home
Search
Downloads
More


10:37
Witty TV Shows
Top Picks for Tascia
Dark TV Dramas
TV Thrillers
Home
Search
Downloads
More


amazon.com
Recommended for You
Amazon.com has new recommendations for you based on items you purchased or told us you own.
The Little Big Things: 163 Ways to Pursue EXCELLENCE
Fascinate: Your 7 Triggers to Persuasion and Captivation
Sherlock Holmes [Blu-ray]
Alice in Wonderland [Blu-ray]

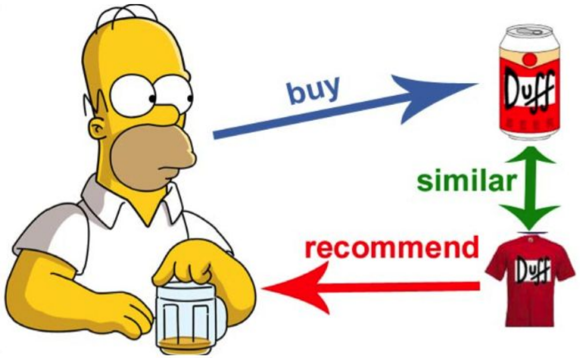

Recommended destinations
We've gathered the best deals in our most popular destinations.
Flic-en-Flac
Mauritius
8 Value Deals
from HK$ 302
Take me there
Balule Game Reserve
South Africa
1 Value deal
from HK$ 504
Take me there
Sorrento
Australia
Previous Stay
2 Value Deals
from HK$ 689
Take me there

Core problem in rec. sys.

- Filter Information for users
- Personalization is the key:
  - Given a certain user, compute the score that quantifies how strongly a user likes item i.

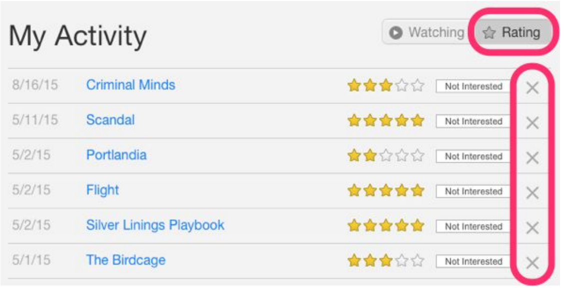
Content-based method

- Define the similarity from items' content
  - Name: cosine similarity
  - Category
  - Rating
  - Description
  - Etc
- Combine them into a final score
- Ranked items based on their similar scores compared to users' purchased item.

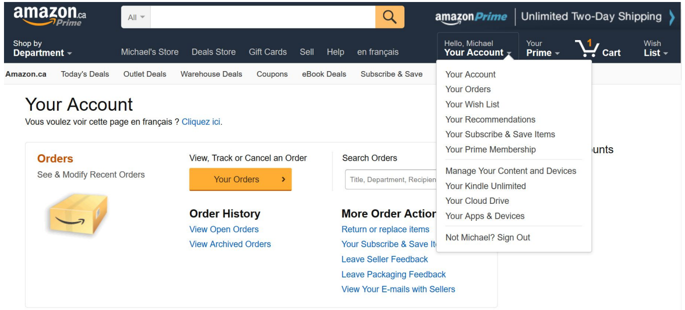


User behaviour

- Content-based methods: only look at the items' information
- The Insights behind the huge interaction behind users and items



Ratings in Netflix



Order History

User-Item matrix

- Content-based methods: only look at the items' information
- The Insights behind the huge interaction behind users and items

Item Vector

User Vector

	Item 1	Item 2	Item 3	…….	Item k-1	Imte k
User 1	1	0	0		3	1
User 2	0	3	1		0	2
...						
User n-1	0	2	0		1	1
User n	0	0	0		0	0

User-based CF

- Find the similarity score between users
- Recommend products which these similar users have liked or bought previously

The rating of item
i given by user v

$$P_{u,i} = \frac{\sum_{v \in U} (r_{v,i} * s_{u,v})}{\sum_{v \in U} s_{u,v}}$$

The similarity between users u and v

The prediction of
an item i for user
u

User Space

$$s_{u,v} = cos(\vec{u}, \vec{v}) = \frac{\vec{u} * \vec{v}}{||\vec{u}|| ||\vec{v}||}$$

Cosine similarity used a lot in information retrieval

Item-based CF

- Find the similarity score between items
- Recommend similar items which were liked or purchased by the users in the past

The rating of item m given by user u

$$P_{u,i} = \frac{\sum_{m \in I} (r_{u,m} * s_{i,m})}{\sum_{m \in I} s_{i,m}}$$

The similarity between items i and m

The prediction of an item i for user u

Item Space

$$s_{i,m} = cos(\vec{i}, \vec{m}) = \frac{\vec{i} * \vec{m}}{||\vec{i}||||\vec{m}||}$$

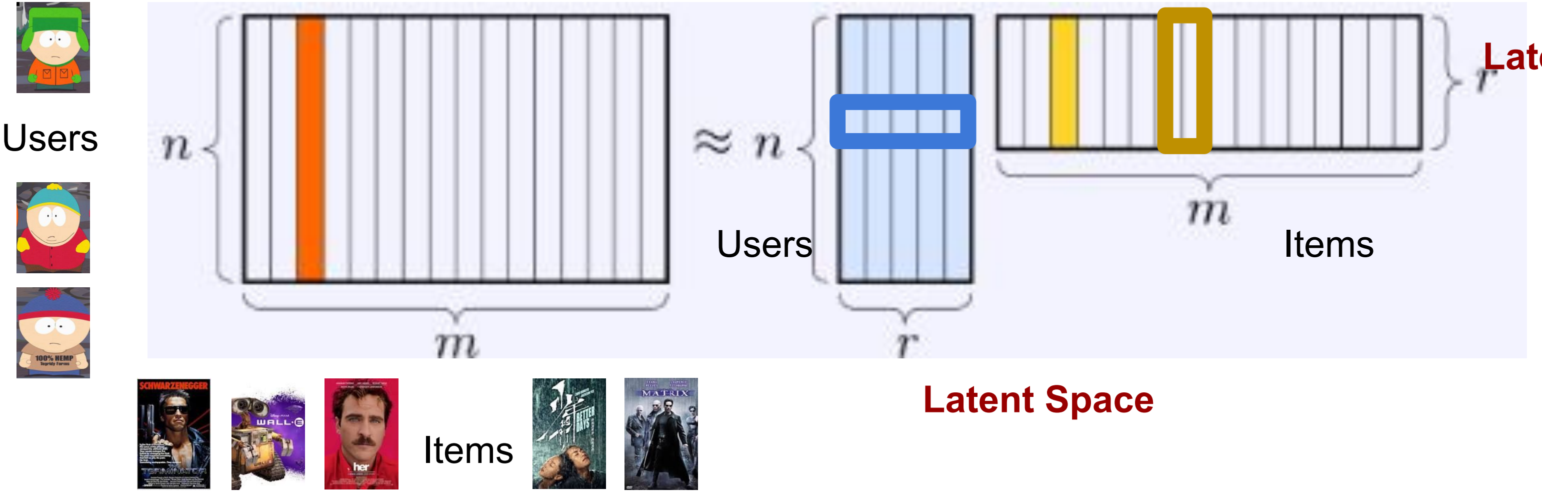
Data sparsity

movieId	1	2	3	4	5	6	7	9	10	11	...	106487	106489	106782	106920	109374
userId																
316	-0.829457	NaN	NaN	NaN	NaN	NaN	-1.329457	NaN	-0.829457	NaN	...	NaN	NaN	NaN	NaN	NaN
320	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
359	1.314526	NaN	NaN	NaN	NaN	1.314526	NaN	NaN	0.314526	0.314526	...	NaN	NaN	NaN	NaN	NaN
370	0.705596	0.205596	NaN	NaN	NaN	1.205596	NaN	NaN	NaN	NaN	...	-1.294404	-0.794404	0.705596	0.205596	NaN
910	1.101920	0.101920	-0.39808	NaN	-0.39808	-0.398080	NaN	NaN	NaN	0.101920	...	NaN	NaN	-0.398080	NaN	NaN

Similarities between users and items are zero

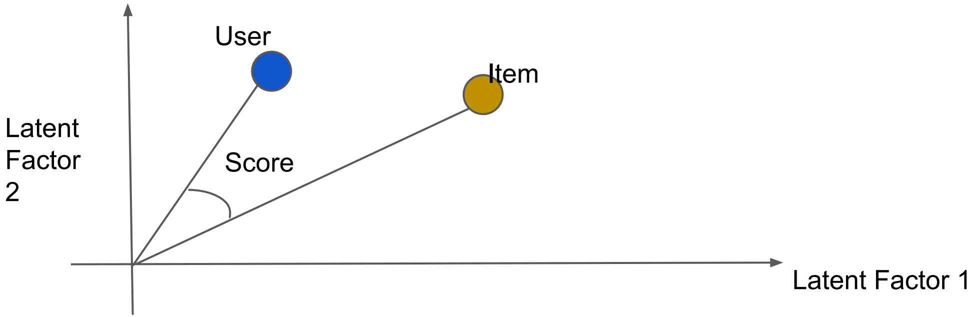
- The core problem behind recommendation sys. is to fill these zero entries, i.e., infer the users preference over the item.
  - Address as data missing problems:
    - Use the mean value of the row
    - Use the mean value of the column
  - Matrix Factorization
    - Singular Value Decomposition
    - Non-Negative Matrix Factorization
    - Auto-encoder

NMF for rec

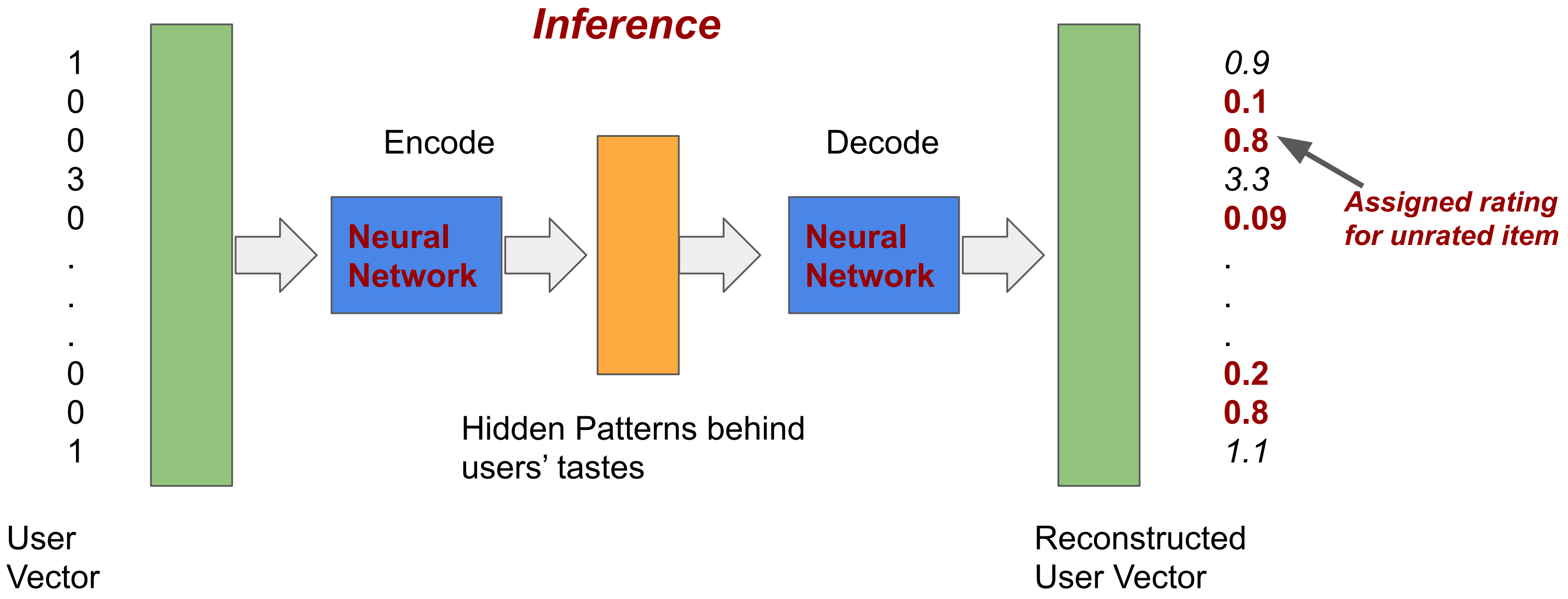


Latent Space

Latent Space



Autoencoder for rec.

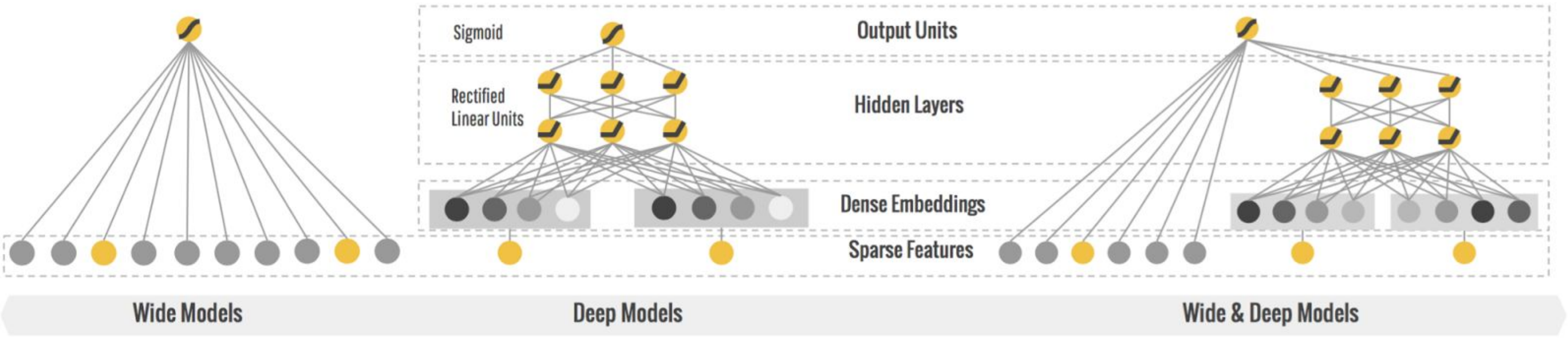


Pros & Cons of CF

- Pros
    - Capture latent users and item factors
    - Can handle sparsity
    - Scalable computation (ALS)
- Cons:
    - Biases (Temporal and Popularity)
    - Cold Start Problem
    - No Context-awareness

Feature-based Methods

Deep & Wide Model from Google



Source: https://ai.googleblog.com/2016/06/wide-deep-learning-better-together-with.html

Feature-based Methods

- Three-class classification problem:
    - Click
    - Impressed but unclick
    - Dislike

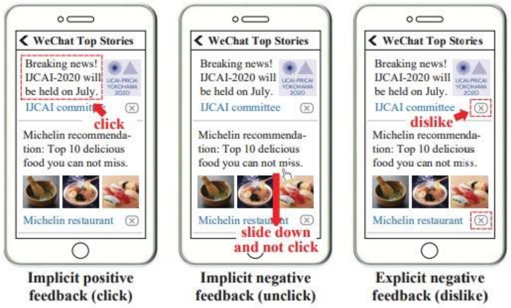
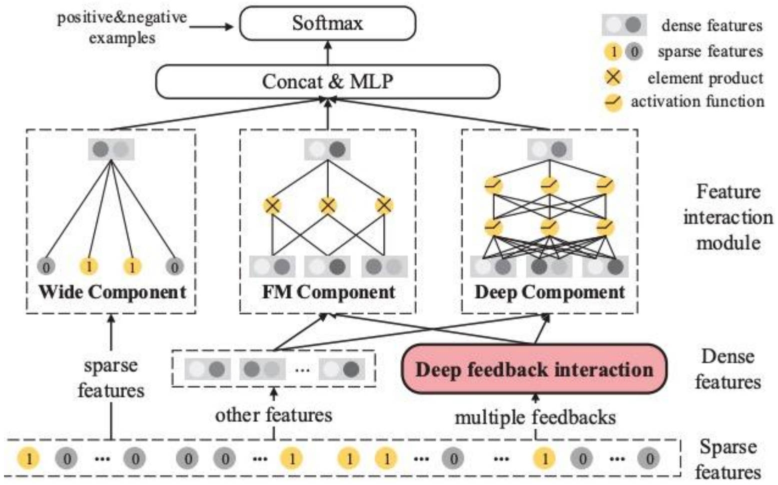


Figure 1: An example of multiple feedbacks in WeChat Top Stories.



(a) Deep feedback network

Source: https://www.ijcai.org/proceedings/2020/0349.pdf

Next Class: Convolutional Neural Network