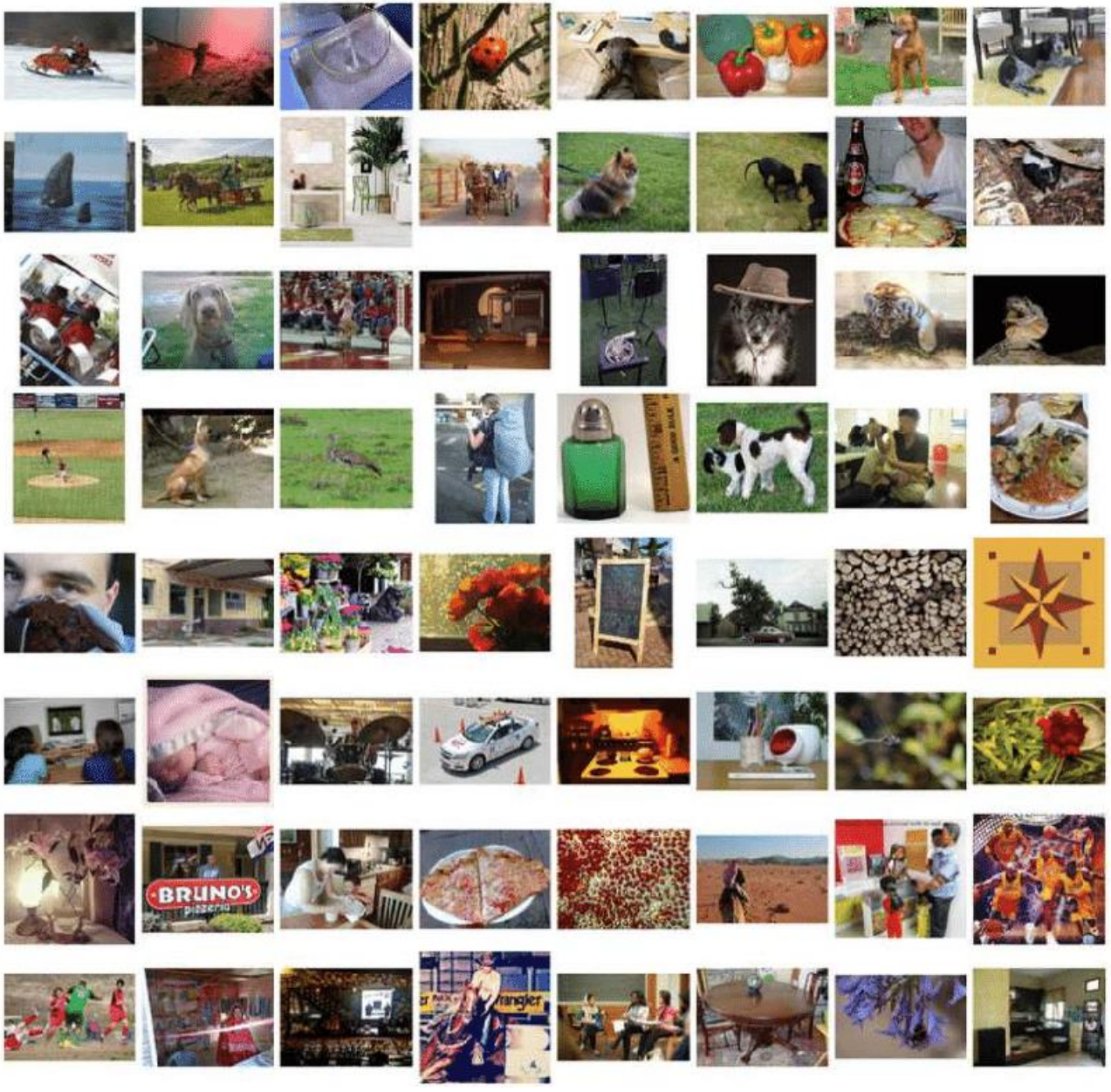
Csapó Tamás Gábor

Deep Learning a gyakorlatban Python és LUA alapon

AutoEncoder

Deep learning híradó (1)

- ImageNet 2012: 14M kép, kézzel címkézve



Deep learning híradó (1)

- Előítéletek…
- ImageNet 2012: 14M kép, kézzel címkézve
- Felügyelet nélküli tanítás, [Steed & Caliskan 2020]
- Eredmény:
  - társadalmi elfogultság
  - pl. fehér ember -> szerszám / néger -> fegyver
  - pl. férfiak -> természettudomány nők -> bölcsészettudomány

- Vajon miért?
  - adatbázis címkézése: emberek -> bias
  - ImageNet: az emberek nagyrészt sztereotip szerepekben
  - nincs tökéletes adatbázis!

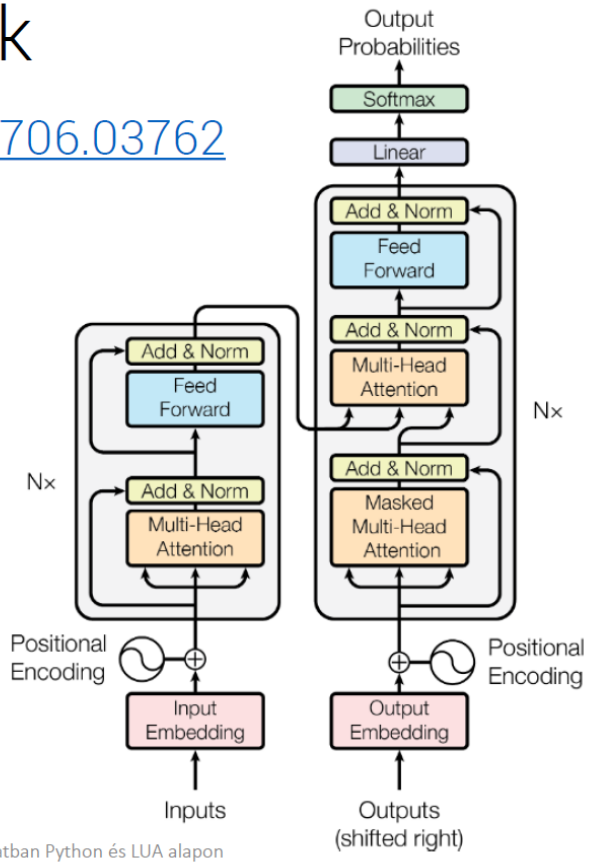
Ryan Steed, Aylin Caliskan, Image Representations Learned With Unsupervised Pre-Training Contain Human-like Biases, https://arxiv.org/abs/2010.15052

Deep learning híradó (2)

- Nagy Transformer hálózatok NLP-ben jól teljesítenek
  - De probléma: óriási számításigény

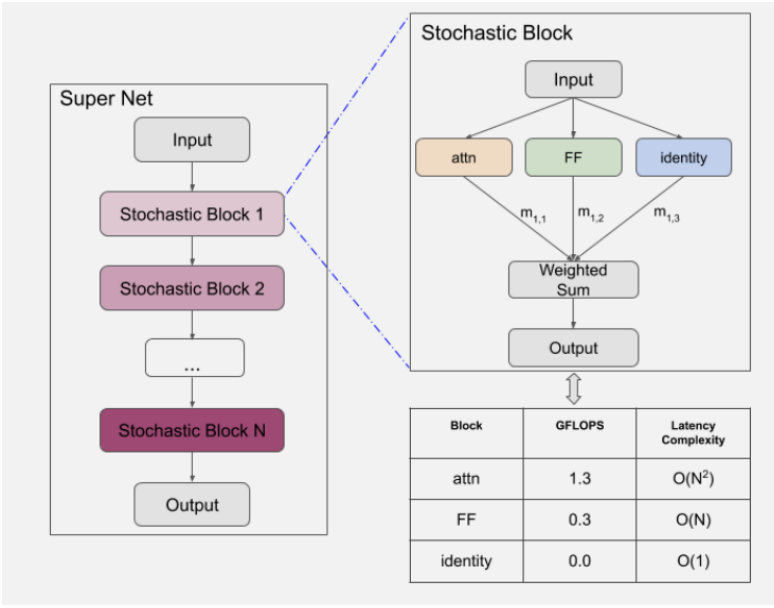
Transzformerek

- https://arxiv.org/abs/1706.03762



Deep learning híradó (2)

- Nagy Transformer hálózatok NLP-ben jól teljesítenek
  - De probléma: óriási számításigény
- Selective Attention
  - „Pay Attention When Required” [Mandava et al. 2020]
  - Transformer-XL-hez és BERT-hez hasonló eredmény, de jóval rövidebb idő alatt
  - Ötlet: néhány self-attention réteg lecserélése sima FF rétegre
  - Hálózat optimalizálása: Neural Architecture Search
  - Eredmény: kb 35%-kal gyorsabb működés



Block	GFLOPS	Latency Complexity
attn	1.3	$O(N^2)$
FF	0.3	O(N)
identity	0.0	O(1)

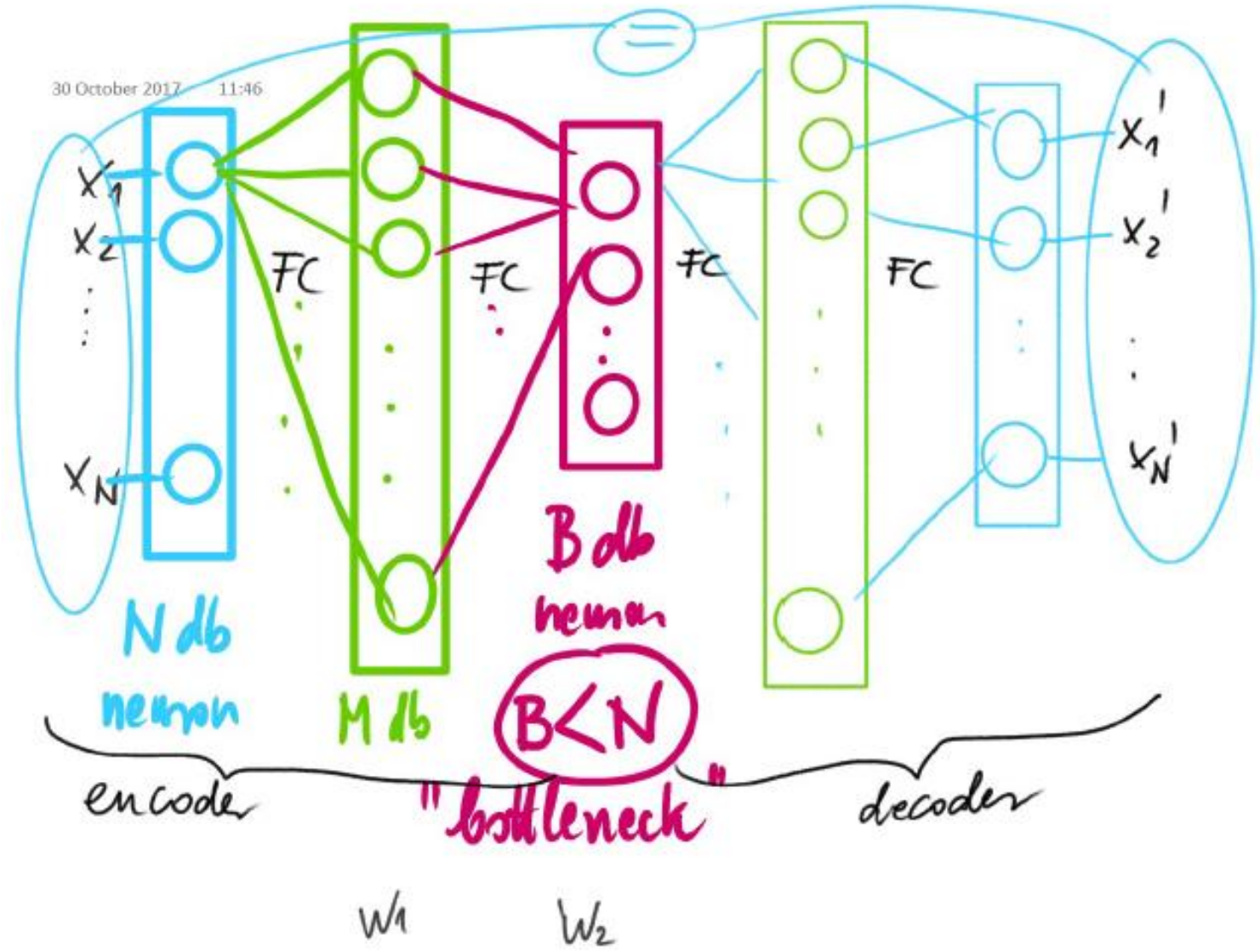
Swetha Mandava, Szymon Migacz, Alex Fit Florea, Pay Attention when Required, https://arxiv.org/abs/2009.04534

Jogi nyilatkozat

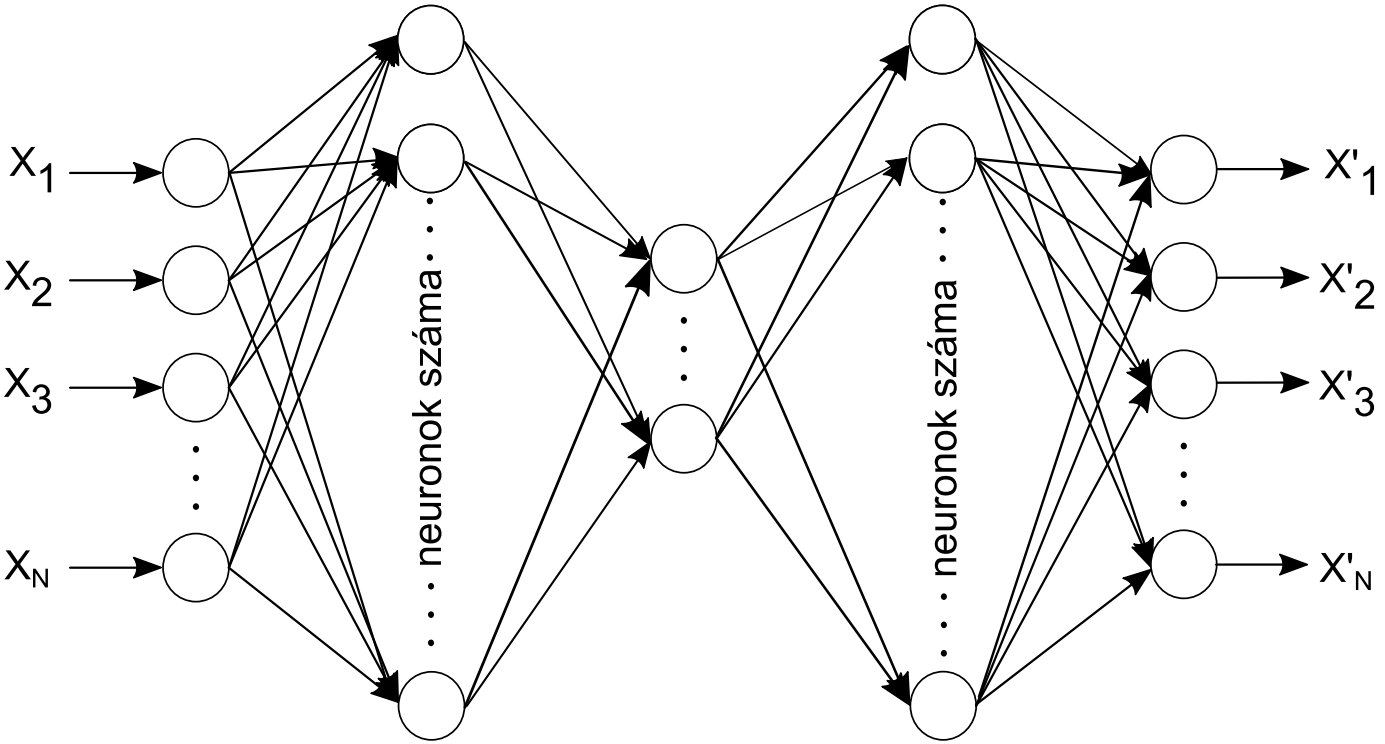
Jelen előadás diái a „*Deep Learning a gyakorlatban Python és LUA alapon*” című tantárgyhoz készültek és letölthetők a http://smartlab.tmit.bme.hu honlapról.

A diák nem helyettesítik az előadáson való részvételt, csupán emlékeztetőül szolgálnak.

AutoEncoder (OneNote)



AutoEncoder



AE implementáció – Keras (1)

```
from keras.layers import Input, Dense
from keras.models import Model

encoding_dim = 32 # 32 / 784 tömörítés

input_img = Input(shape=(784,))
# "encoded": a bemenet tömörített változata
encoded = Dense(encoding_dim,
      activation='relu')(input_img)
# "decoded": a bemenet veszteséges visszaállítása
decoded = Dense(784, activation='sigmoid')(encoded)
# bemenet és visszaállított változata
autoencoder = Model(input_img, decoded)
```

AE implementáció – Keras (2)

```
autoencoder = Model(input_img, decoded)


# "encoder" modell: bemenetből tömörített
encoder = Model(input_img, encoded)
encoded_input = Input(shape=(encoding_dim,))
decoder_layer = autoencoder.layers[-1]


# "decoder" modell: visszaállítás
decoder = Model(encoded_input,
decoder_layer(encoded_input))


autoencoder.compile(optimizer='adadelta',
     loss='binary_crossentropy')
```

AE implementáció – Keras (3)

```
autoencoder.fit(x_train, x_train,
        epochs=50,
        batch_size=256,
        shuffle=True,
        validation_data=(x_test, x_test))


# néhány digiten encode és decode
encoded_imgs = encoder.predict(x_test)
decoded_imgs = decoder.predict(encoded_imgs)
```

AE implementáció – eredmény

- Felül: eredeti képek
- Alul: visszaállított képek

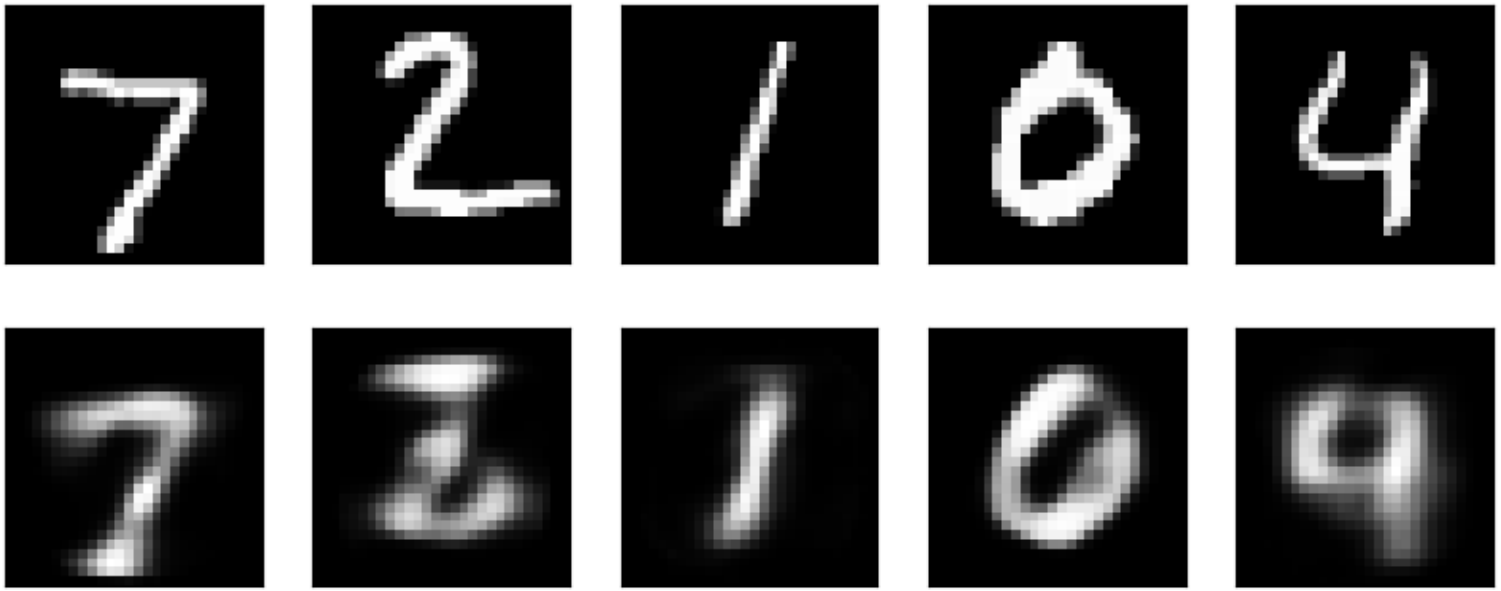
AE – bottleneck mérete

- `encoding_dim = 16 # rejtett réteg ('bottleneck')`

- Felül: eredeti képek
- Alul: visszaállított képek

AE – bottleneck mérete

- `encoding_dim = 8 # rejtett réteg ('bottleneck')`



- Felül: eredeti képek
- Alul: visszaállított képek

AE: mire jó?

- Adat tömörítés
  - Dimenzió csökkentés
    - MNIST 784 pixel helyett elég 16 szám (kb. 2%)
  - Adat-specifikus tömörítési mód
  - Veszteséges tömörítés (mint a JPEG vagy MP3)
  - A tömörítést automatikusan, adatok alapján tanulja

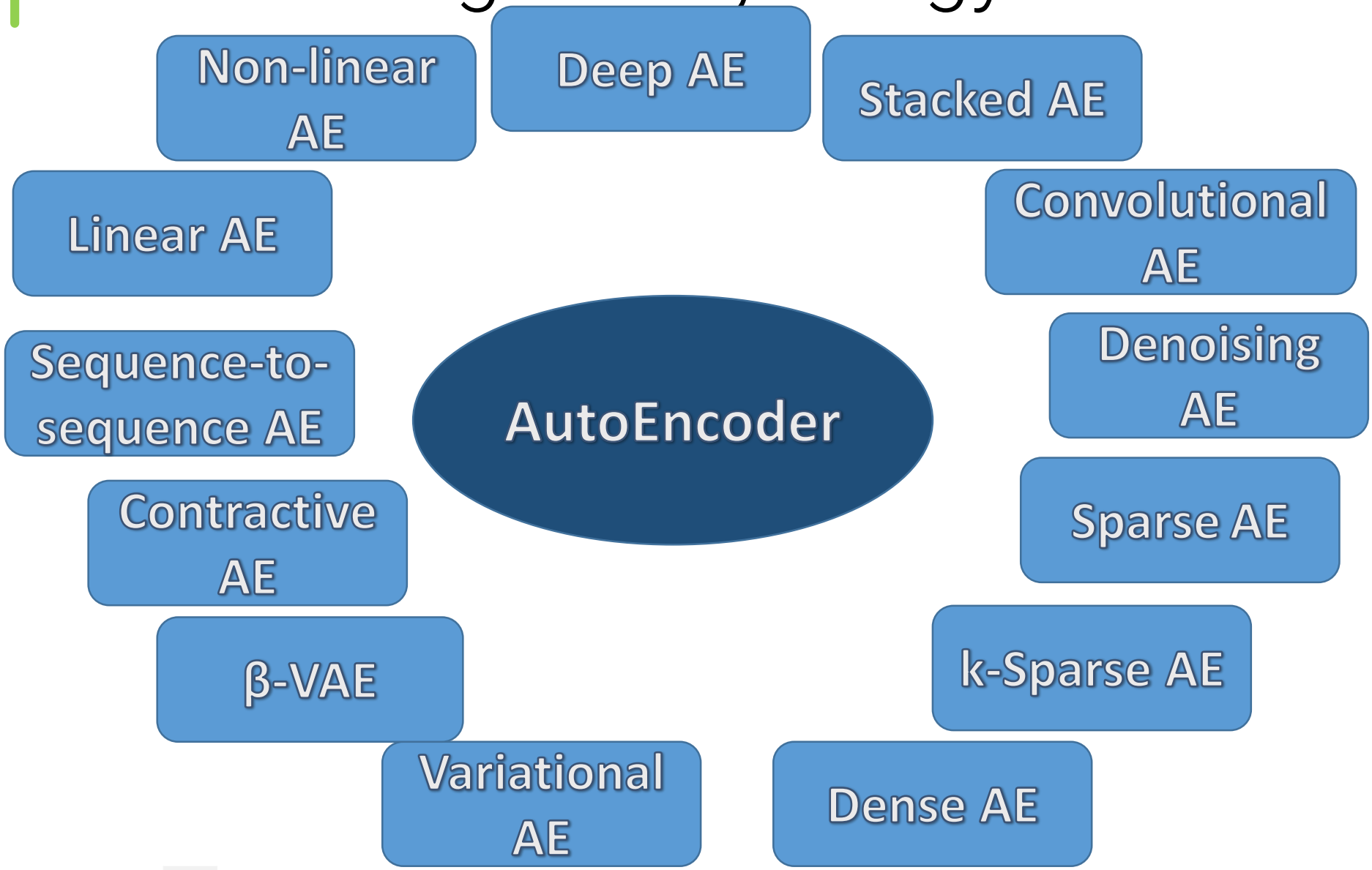

THAT'S NOT ENOUGH
WE HAVE TO GO DEEPER

AE: mire jó?

- Adat tömörítés
  - Dimenzió csökkentés
    - MNIST 784 pixel helyett elég 128 szám
  - Adat-specifikus tömörítési mód
  - Veszteséges tömörítés (mint a JPEG vagy MP3)
  - A tömörítést automatikusan, adatok alapján tanulja

- Igazából nem jó adat tömörítésre
  - Képfeldolgozásban jobb a JPEG
  - Zenefeldolgozásban jobb az MP3
  - Túlságosan adat-specifikus

… azért mégsem ilyen egyszerű



Csapó Tamás Gábor

Deep Learning a gyakorlatban Python és LUA alapon

Felügyelet nélküli tanítás AutoEncoder-rel

Felügyelt tanulás

- Supervised learning, ellenőrzött tanulás

- Adatok: input + output
- Cél: input alapján output becslésére
  gépi tanuló rendszer
- Regresszió vagy osztályozás

- De mi van, ha nem áll rendelkezésre output?

Felügyelet nélküli tanulás

- Unsupervised learning, nemellenőrzött tanulás

- Adatokon hasonlóságok megállapítása
  - „Hasonló” tulajdonsággal rendelkező minták keresése
- Adatok klaszterezése (=klaszteranalízis)
  - Osztályok előre nem ismertek
  - Osztályok kialakítása is feladat
  - pl. K-means algoritmus
- Adattömörítés
  - Adatok leírása kevesebb paraméterrel
  - pl. PCA algoritmus
- Mély hálózatok előtanítása
  - „Pretraining”

K-means (K-közép) algoritmus

- Előre megadjuk a klaszterek számát: K
- Véletlenszerűen létrehozunk K db klasztert és K db klaszter középpontot
- Minden pontot abba a klaszterbe sorolunk, amelynek a középpontjához legközelebb helyezkedik el
- Kiszámoljuk az új klaszter középpontot
- Leállási feltételig iterálunk

- (MacQueen, 1967)

K-means (K-közép) algoritmus

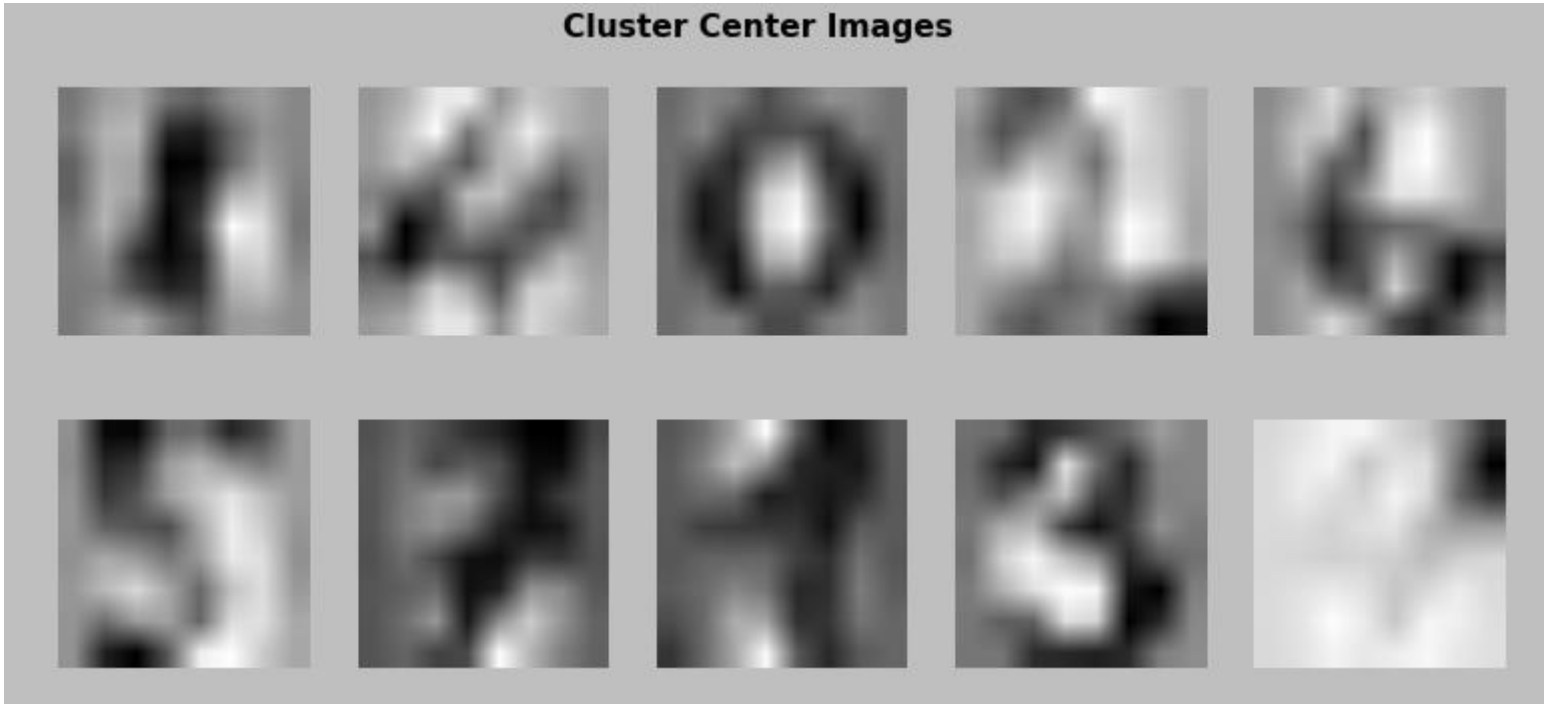
```
# a K-means algoritmus a 'cluster' modulban van
from sklearn import cluster


# K-means modell létrehozása
clf = cluster.KMeans(init='k-means++',
n_clusters=10, random_state=42)


# klaszteranalízis elindítása
clf.fit(X_train)
```

K-means (K-közép) algoritmus

- `digits' adatbázis, K = 10
- Klaszter közepek

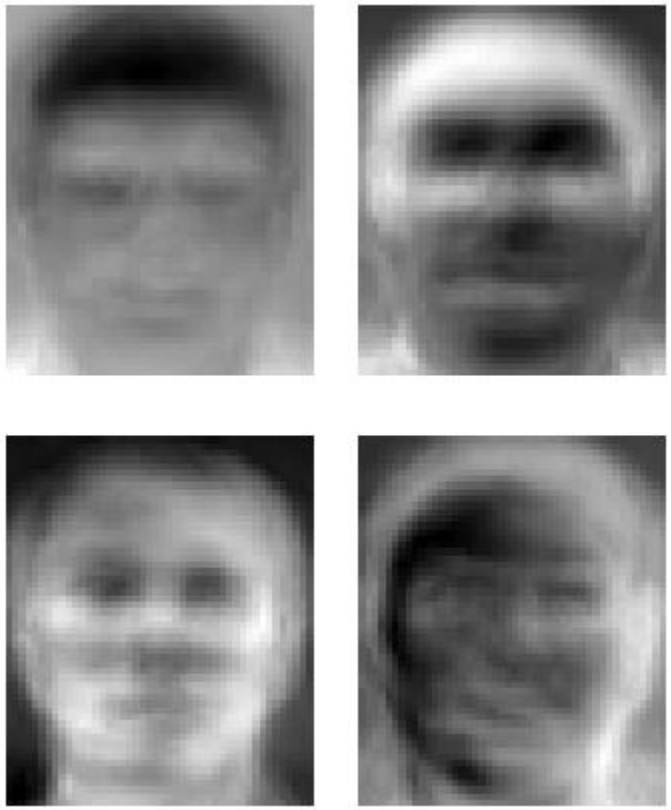


Forrás: https://www.datacamp.com/community/tutorials/machine-learning-python#kmeans

Principal Component Analysis

- PCA
- Főkomponens-analízis
- Dimenzió csökkentés, pl. adat vizualizációhoz
- „n-dimenziós ellipszoid” illesztése az adatokhoz
- (Pearson, 1901)

PCA - EigenFaces



- Sirovich and Kirby (1987)

Forrás: https://en.wikipedia.org/wiki/Eigenface

Principal Component Analysis

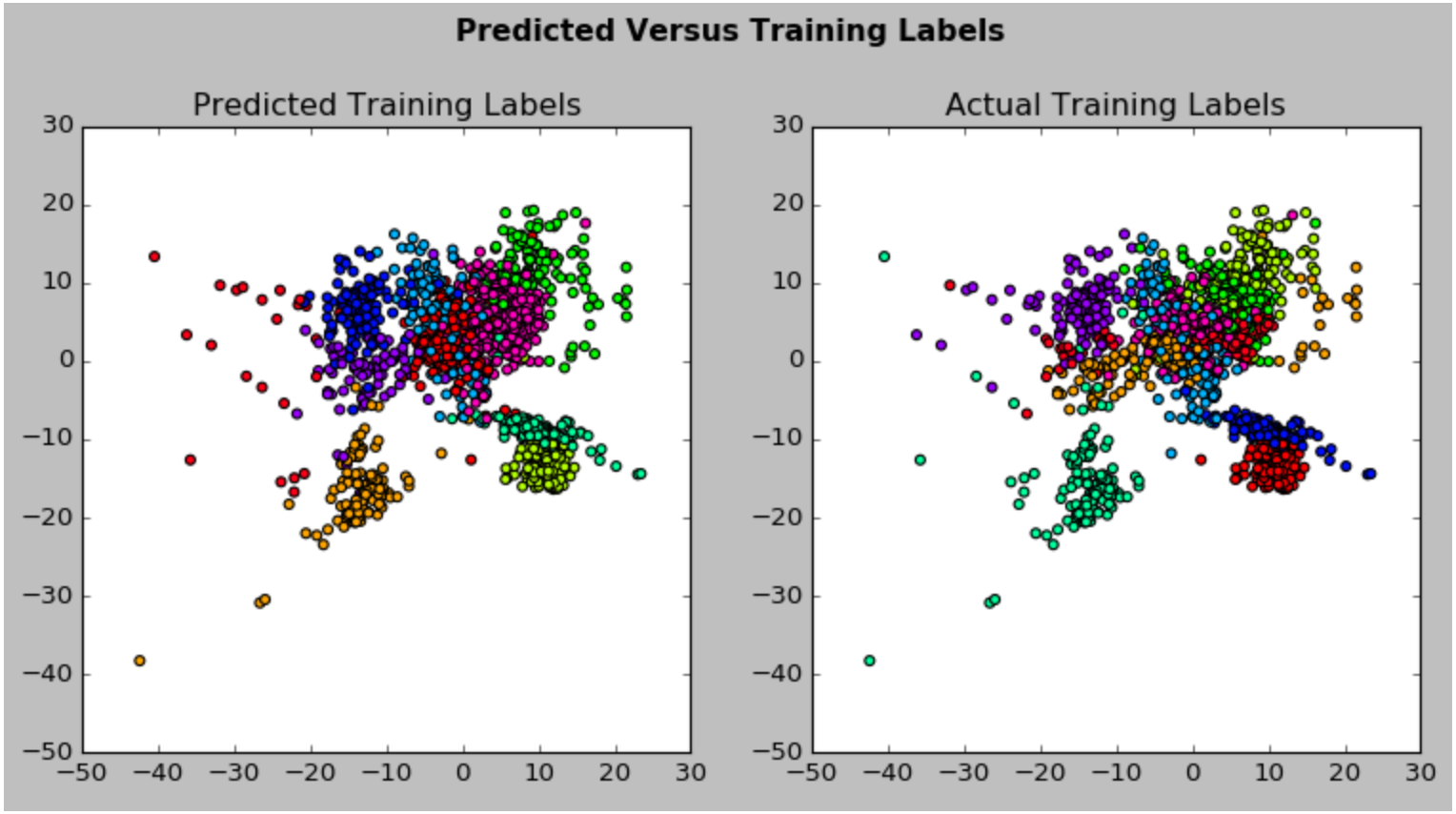
```
# a PCA algoritmus a 'decomposition' modulban
from sklearn.decomposition import PCA


# PCA modell és elindítás
X_pca = PCA(n_components=2).
      fit_transform(X_train)


# klaszter közepek kiszámolása
# és minden csoporthoz klaszter index
clusters = clf.fit_predict(X_train)
```

K-means és PCA algoritmus

- `digits' adatbázis 2D reprezentációja
- Klaszterekre bontás pontossága



Forrás: https://www.datacamp.com/community/tutorials/machine-learning-python#kmeans

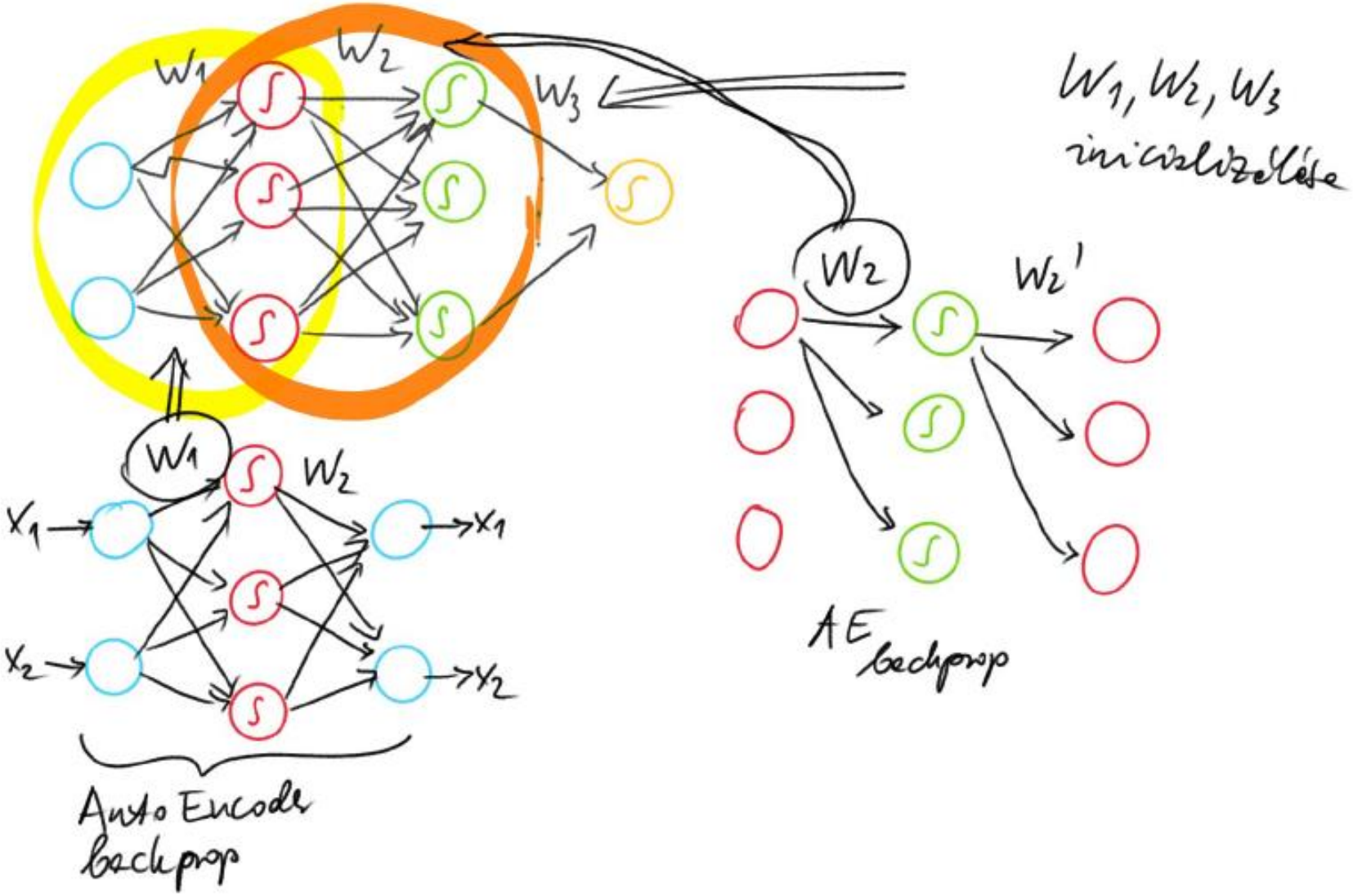
Pretraining / előtanítás (1)

- Hagyományos inicializálás
  - Súlyok inicializálása véletlen számokkal
- De: sok rejtett réteg hatékony tanításához ez nem megfelelő
  - Az alsóbb és felsőbb rétegeken a súlyok nagyságrendje különböző
  - Az alsóbb és felsőbb rétegek egyidejű finomhangolása szükséges

Pretraining (2)

- Hagyományos véletlen inicializálás helyett
- Háló előtanítása, (Hinton&, 2006)
  - Rétegenként, felügyelet nélküli tanító algoritmusokkal
  - Könnyebb feladat, mint egy teljes ellenőrzött (supervised) tanítás
  - Pl. Deep Belief Network, AutoEncoder, (Bengio&, 2007)
  - Felügyelet nélküli előtanítás után háló tanítása hagyományos módon, backprop-pal

Pretraining (3) (OneNote)



Pretraining (4)

- Erhan et al., „Why Does Unsupervised Pre-training Help Deep Learning?”, JMLR, 2010, http://www.jmlr.org/papers/volume11/erhan10a/erhan10a.pdf

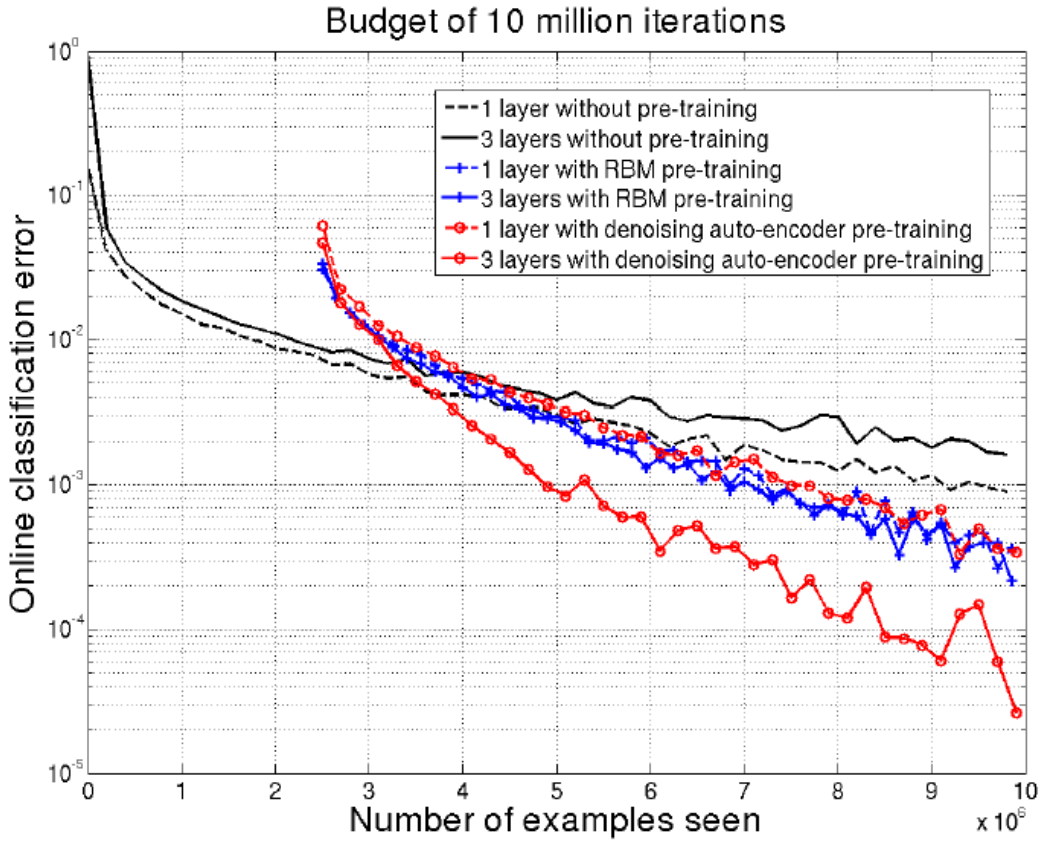
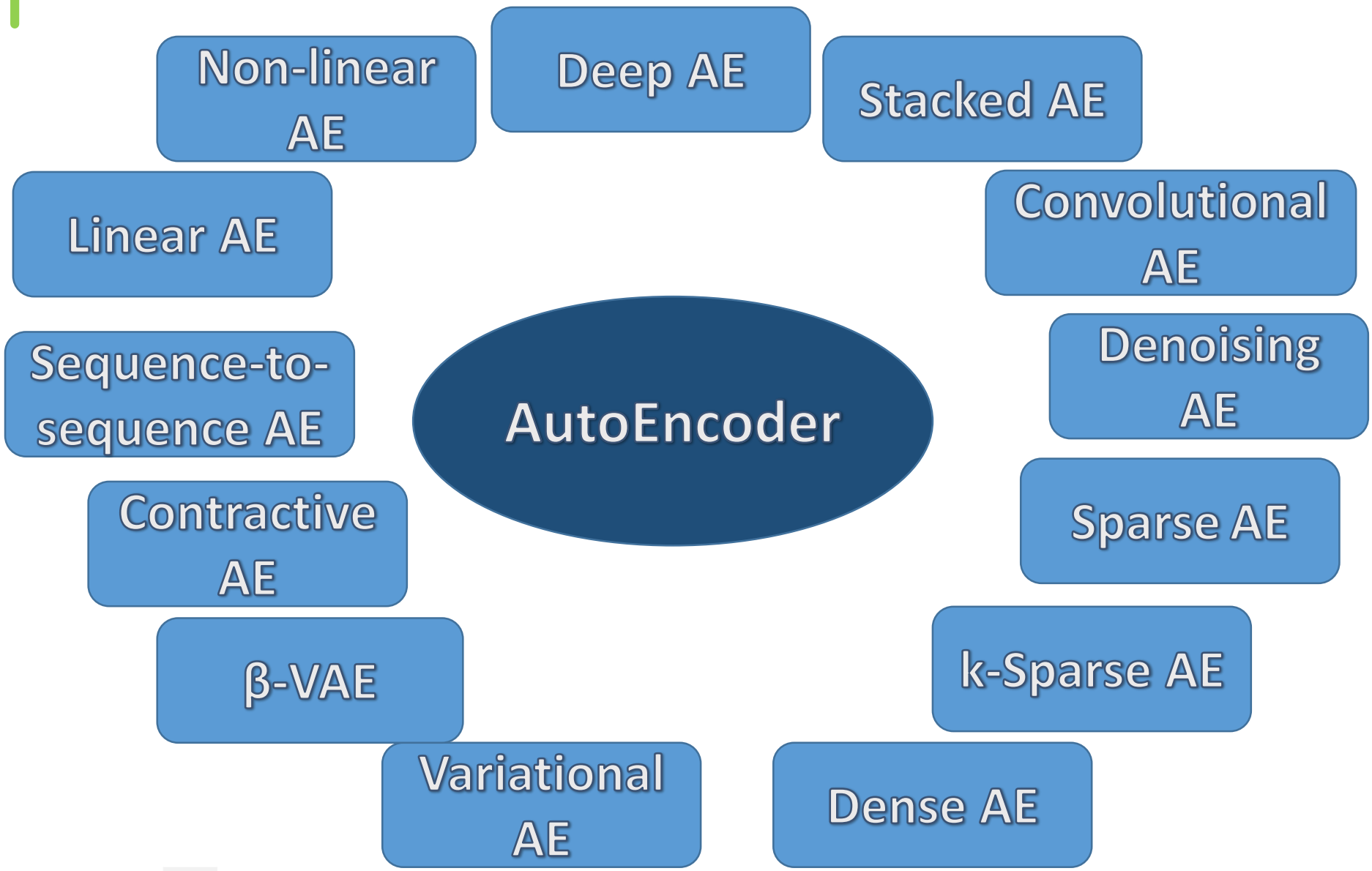


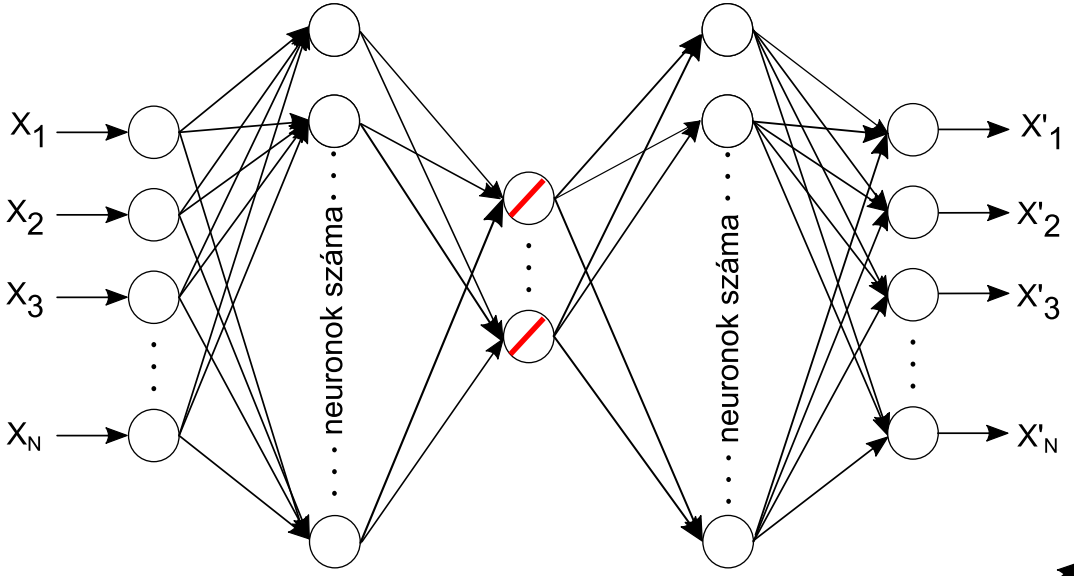
Figure 11: Comparison between 1 and 3-layer networks trained on `InfiniteMNIST`. Online classification error, computed as an average over a block of last 100,000 errors.

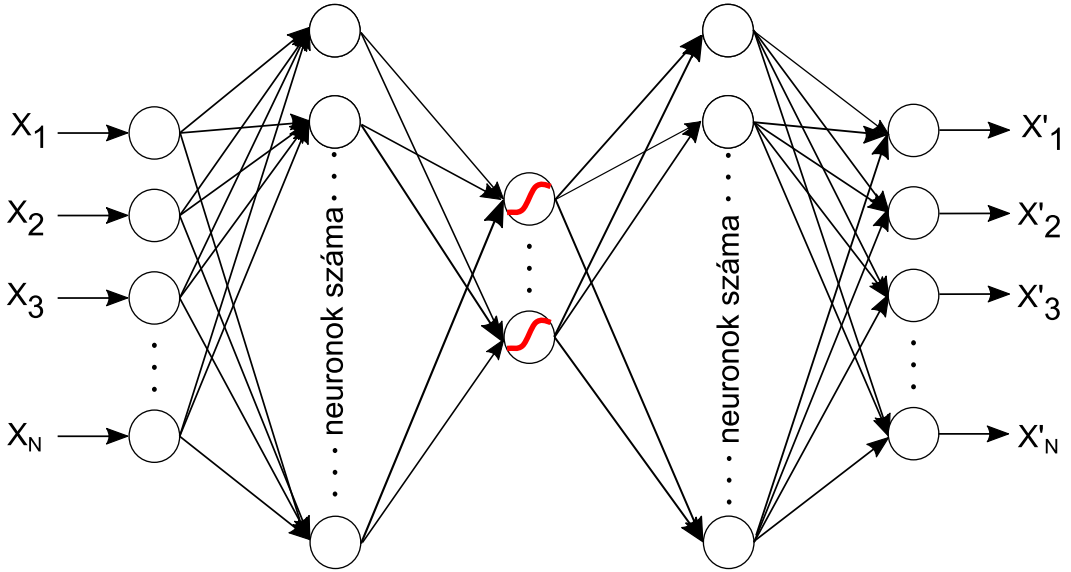
Pretraining (5)

- Ma már tipikusan nem használunk előtanítást
  - Időigényes
  - További extra hiperparaméterek
  - ReLU és DropOut miatt szükségtelen
  - Ha sok adatunk van, akkor nem kell
  - Gyorsabb GPU-k, mint 10-15 éve

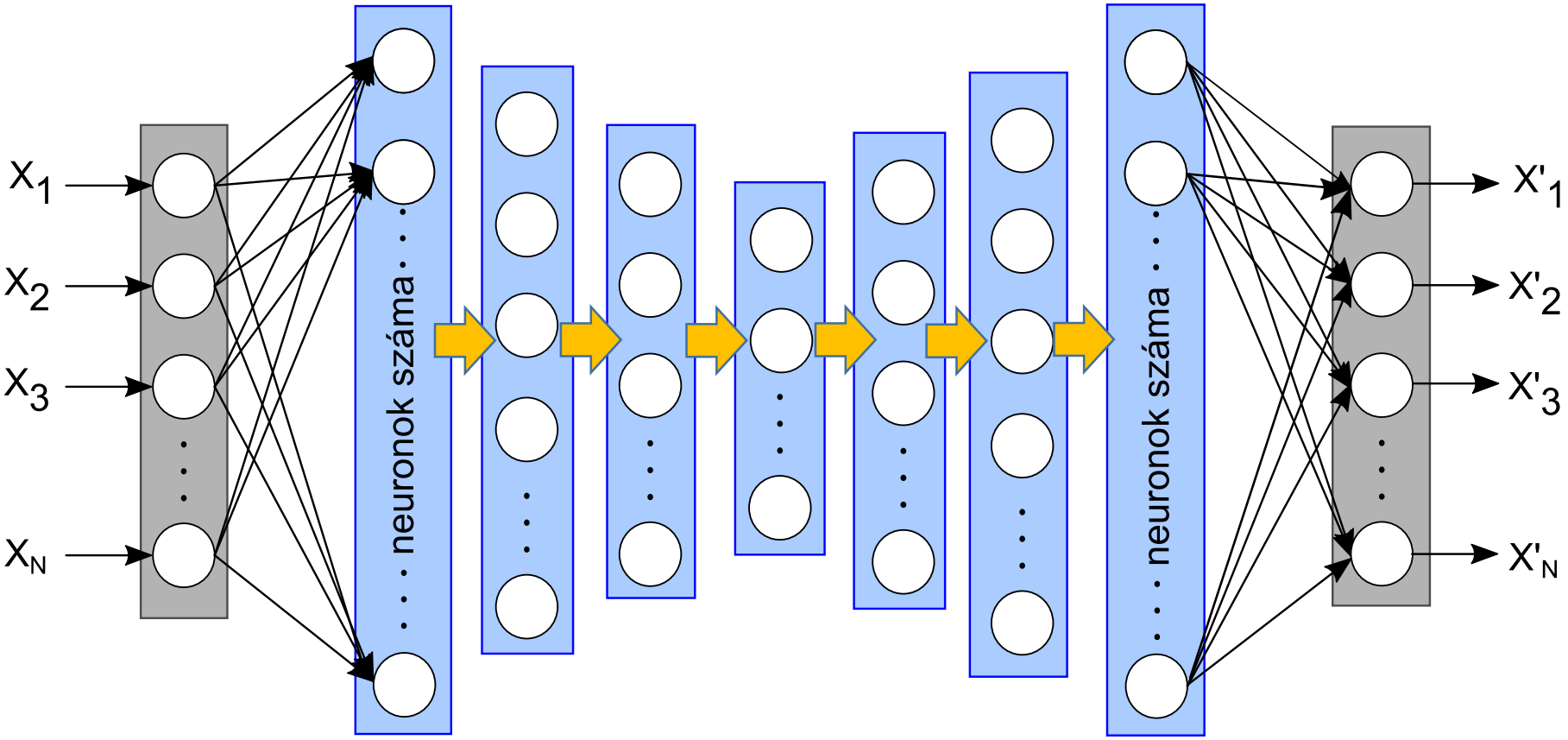

AutoEncoder
Deep AE
Non-linear AE
Stacked AE
Linear AE
Convolutional AE
Sequence-to-sequence AE
Denoising AE
Contractive AE
Sparse AE
β-VAE
k-Sparse AE
Variational AE
Dense AE

Linear vs. non-linear AE (1)

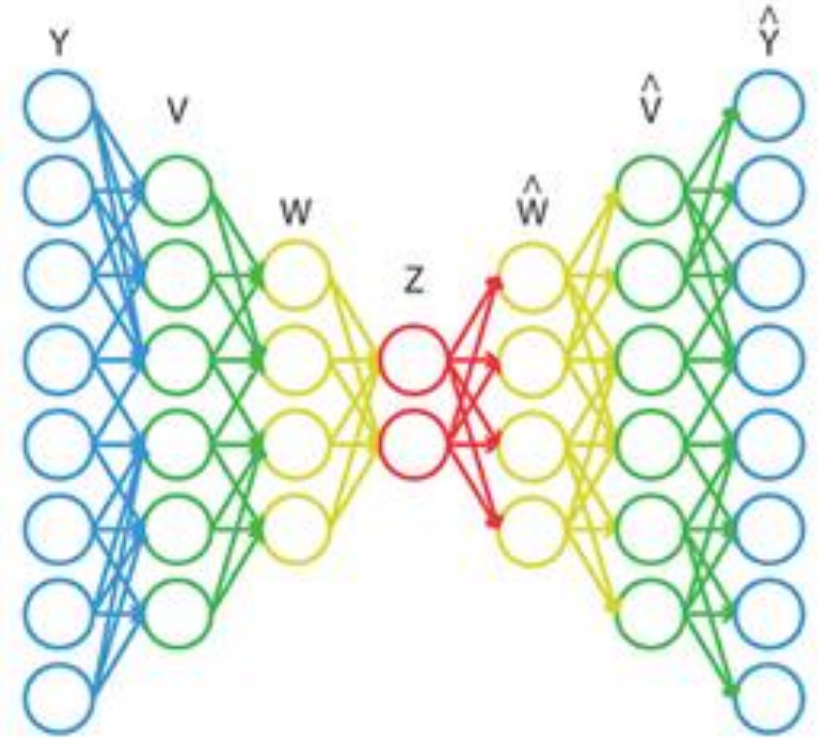




Deep AE

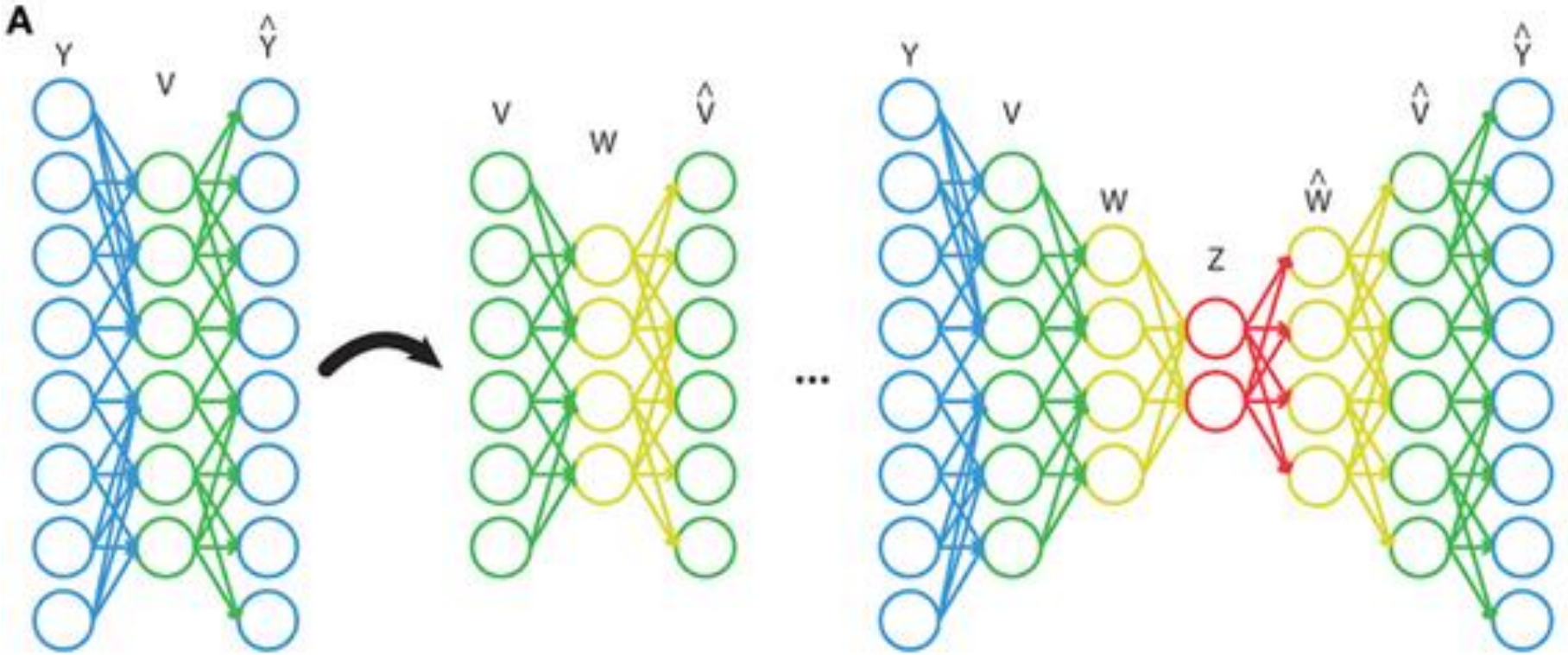


Stacked AE (1)



Forrás: https://www.researchgate.net/profile/Konrad_Kording/publication/274728436/figure/fig2/AS:271715934666753@1441793535194/Figure-2-A-A-stacked-autoencoder-is-trained-on-high-dimensional-data-im-i-1.png

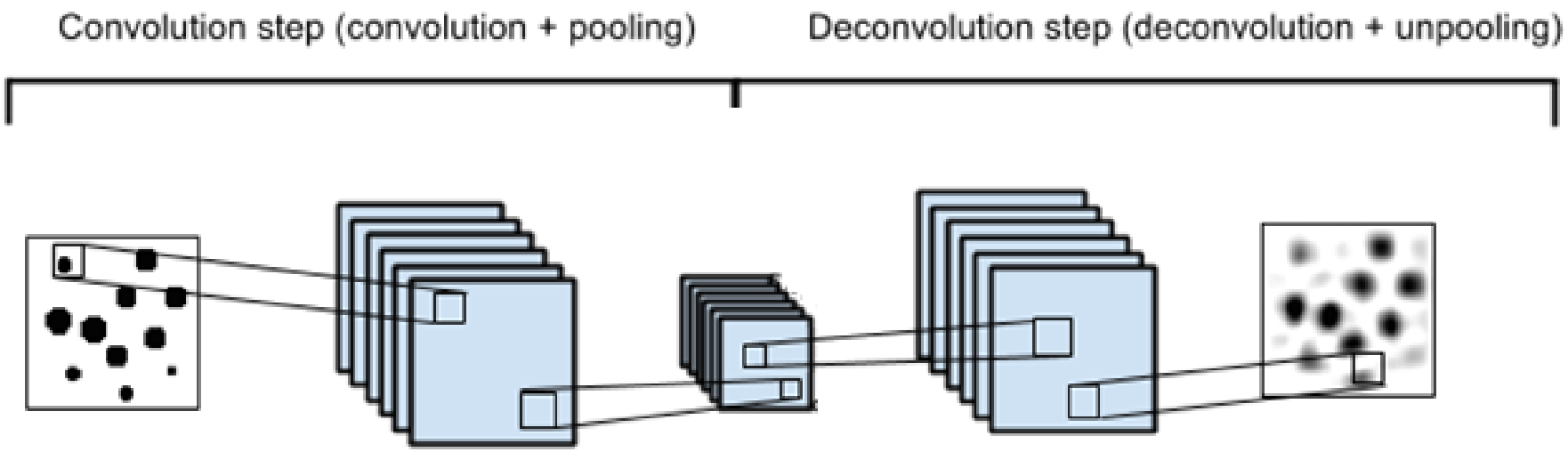
Stacked AE (2)



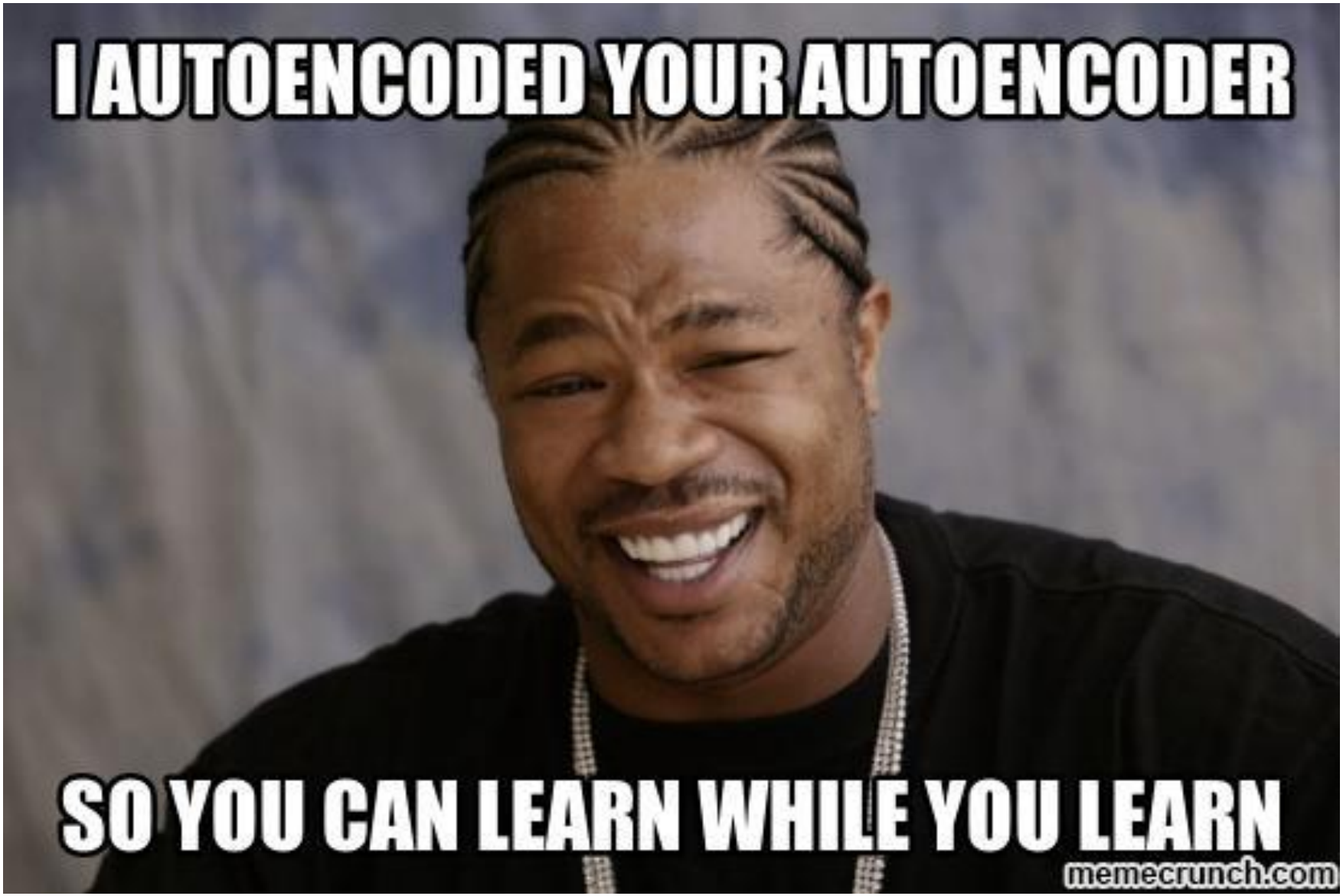
Forrás: https://www.researchgate.net/profile/Konrad_Kording/publication/274728436/figure/fig2/AS:271715934666753@1441793535194/Figure-2-A-A-stacked-autoencoder-is-trained-on-high-dimensional-data-im-i-1.png

Convolutional AutoEncoder

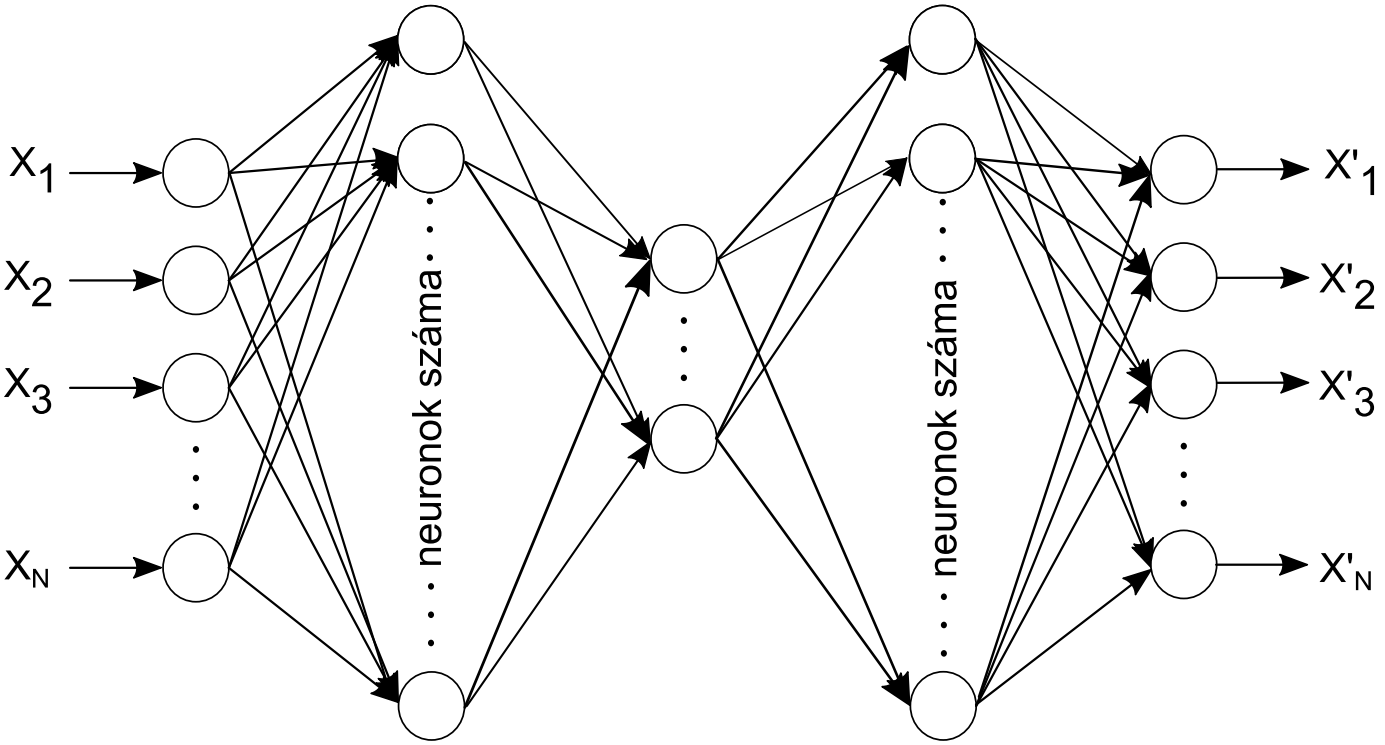
- Képek osztályozására CNN jól működik
- Ötlet: CNN encoder + (FC) + CNN decoder

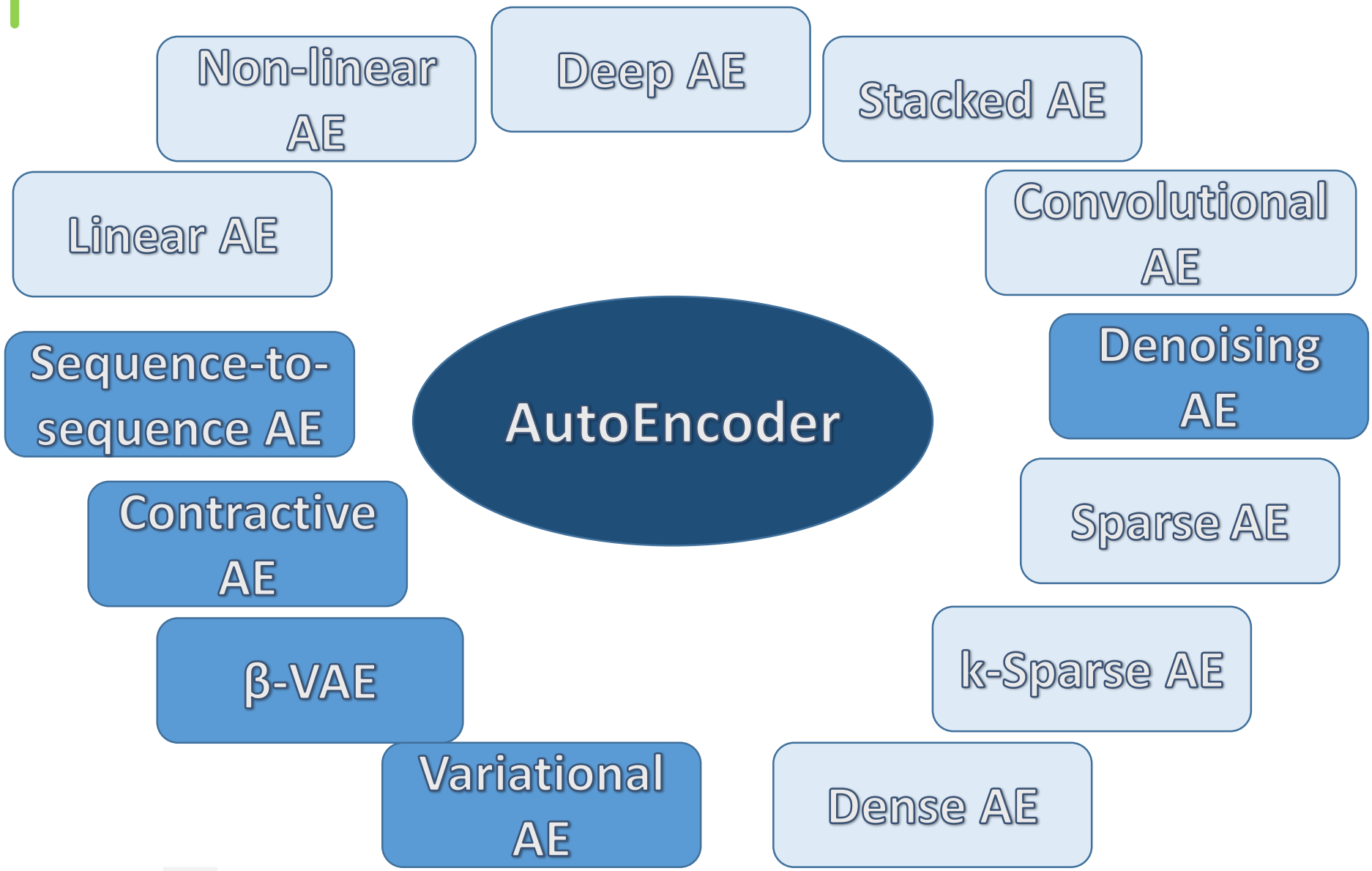


Forrás: https://swarbrickjones.wordpress.com/2015/04/29/convolutional-autoencoders-in-pythontheanolasagne/

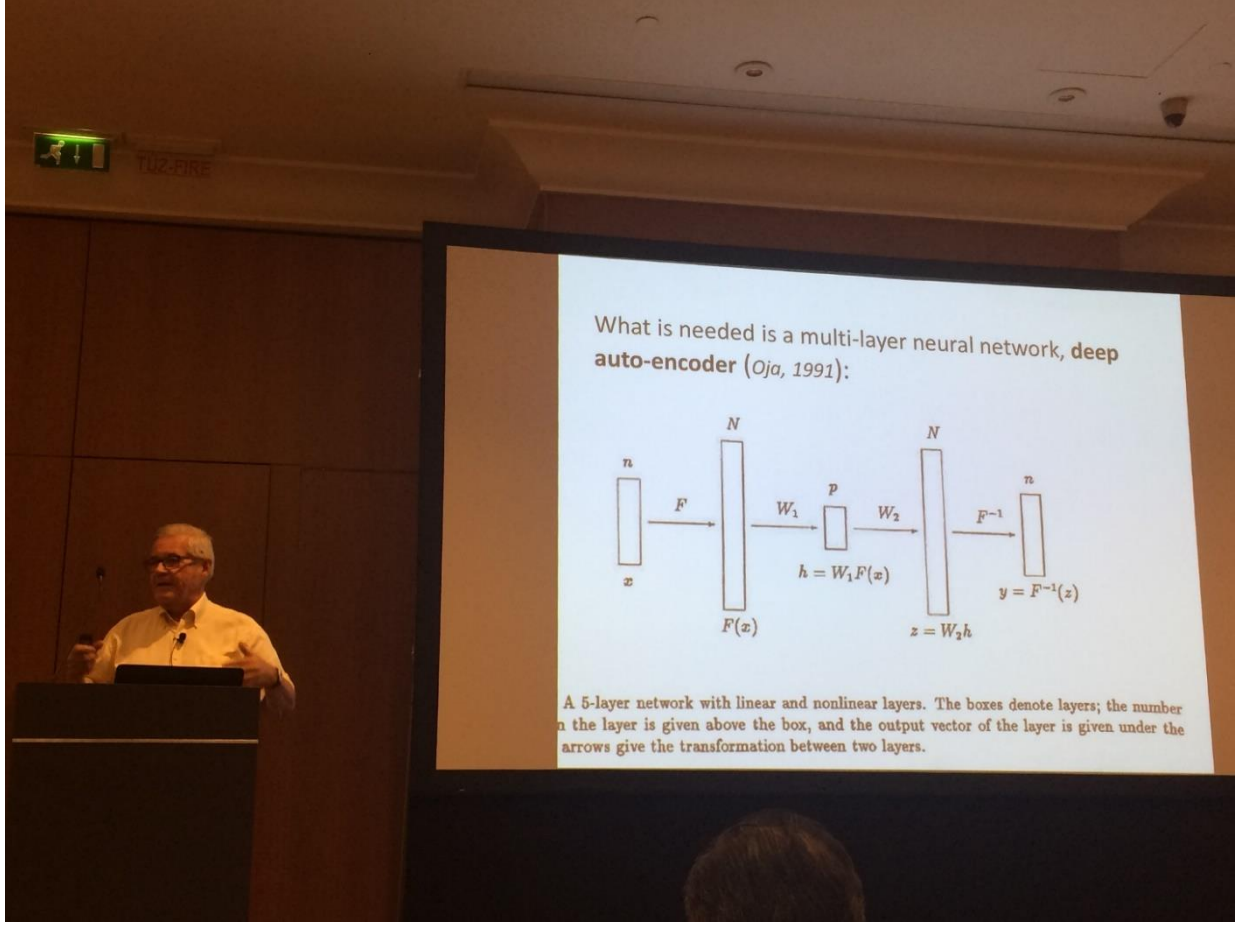

I AUTOENCODED YOUR AUTOENCODER
SO YOU CAN LEARN WHILE YOU LEARN
memecrunch.com

Denoising AE




Non-linear AE
Deep AE
Stacked AE
Linear AE
Convolutional AE
Sequence-to-sequence AE
AutoEncoder
Denoising AE
Contractive AE
Sparse AE
β-VAE
k-Sparse AE
Variational AE
Dense AE

AutoEncoder: új ötlet?



- Erkki Oja demonstrated in 1982 that a neural network with a linear activation function essentially learns the principal component representation of the input data.
- E. Oja, “Simplified neuron model as a principal component analyzer,” Journal of mathematical biology, vol. 15, no. 3, pp. 267–273,

Köszönöm a figyelmet!

csapot@tmit.bme.hu

SmartLab
Intelligent Interactions

http://smartlab.tmit.bme.hu

NVIDIA
GPU
EDUCATION
CENTER

Csapó Tamás Gábor

Deep Learning a gyakorlatban Python és LUA alapon

AutoEncoder gyakorlat

http://bit.ly/DL-GYAK11-csapot

AE és VAE: mire jó?

- Adat tisztítás
  - Bemenet: zajos adat
  - Kimenet: tiszta adat
- Klaszterezés, felügyelet nélküli tanulás
  - Az eddig tanult NN típusok erre nem alkalmasak!
- Dimenzió csökkentés
  - Adat vizualizációhoz
- Anomália detekció
  - Új adat tulajdonságai mennyire különböznek az eddigiektől?
- Adat generálás
  - VAE: generatív modell