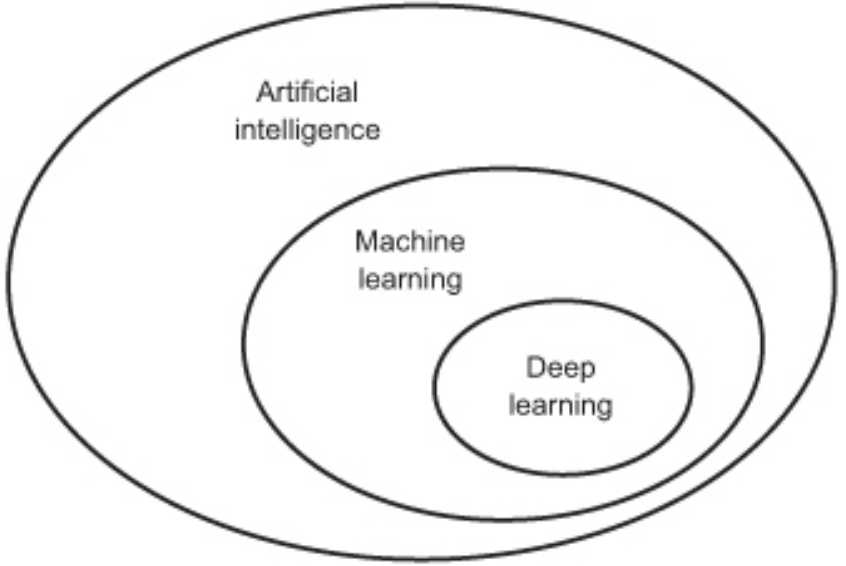
TAITOTALO

RNN
(Recurrent Neural Networks)

Neuroverkot

- Saaneet inspiraation ihmisen aivoista ja neuroneista
- Koneellinen neuroverkko on yksinkertaisempi kuin ihmisaivot
- Neuroverkot keskittyvät yhden asian ratkaisemiseen
- Perustuvat laskentaan

Figure 1.1. Artificial intelligence, machine learning, and deep learning

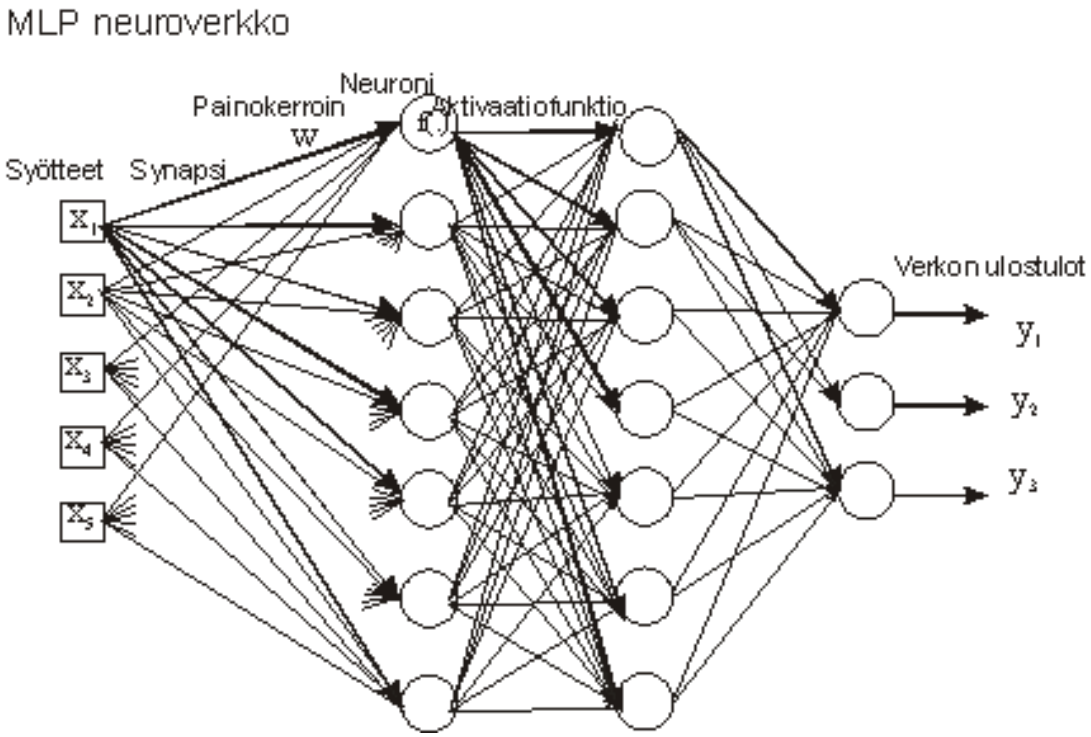


Deep Learning with Python

Neuroverkot

- Neuroverkko on tekoälyn toimintamalli, joka jäljittelee ihmisen aivojen toimintaa.
- Neuroverkot ovat osa syväoppimista, ja ne perustuvat solmujen (node) verkostoon, jotka muodostavat monimutkaisen rakenteen.
- Solmut vastaavat ihmisaivojen neuroneja ja välittävät tietoa toisilleen.
- Syvät neuroverkot koostuvat useista kerroksista, jotka kommunikoivat keskenään.
- Neuroverkkoja käytetään sovelluksissa, kuten:
  - kuvantunnistus
  - luonnollinen kielenkäsittely
  - puheentunnistus

Neuroverkot



http://avoinelama.fi/hingo/filosofia/kollektiivinenajattelu.html

- Syötteet (input)
- Piilossa olevat tasot (hidden layers) ja neuronit niissä
- Tulos(te) (output)

- Lisäksi
  - Painokertoimet (weights)
  - Aktivaatiofunktiot (activation)
  - Harha (bias)

Esimerkiksi luokittelussa malli opetetaan opetusjoukolla, jossa syötteenä annetun havainnoin (x) luokka on jo tiedossa (y)

IoT TAITOTALO

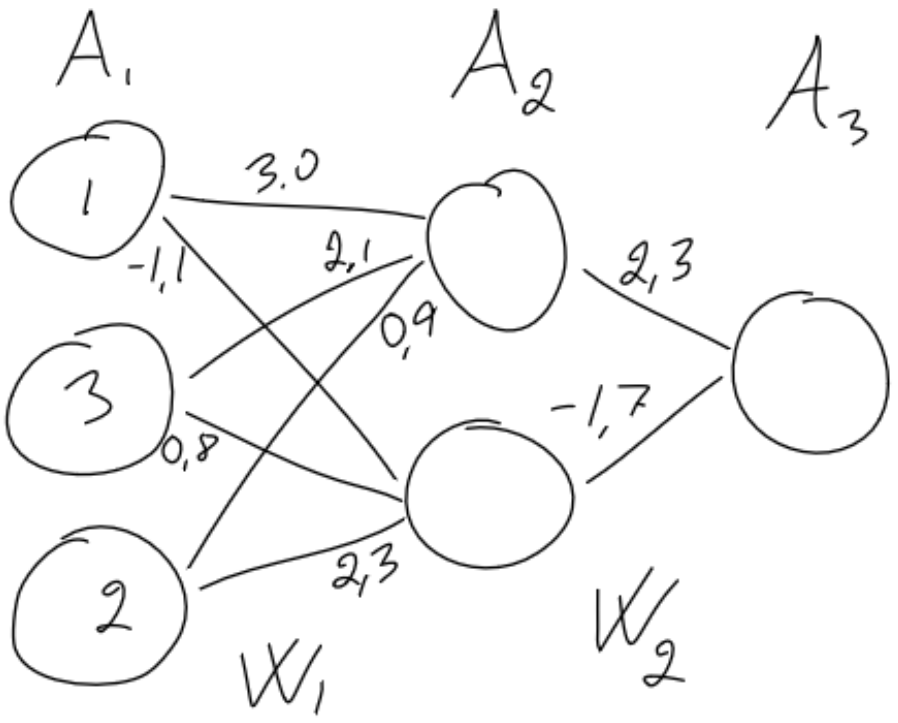
Laske

2. We have the following small network, where the weights are written beside the edges, and there are no bias terms.

   The values of the cells are stored in matrices A_1, A_2 and A_3, where

$$A_1 = \begin{pmatrix} 1 \\ 3 \\ 2 \end{pmatrix}, \quad A_2 = W_1 A_1, \quad A_3 = W_2 A_2.$$

   a) What are the matrices W_1 and W_2?
   b) Calculate the matrices A_2 and A_3.



RNN (Recurrent Neural Networks)

- RNN käsittelee dataa aikaperustaisesti tai sekvenssimuotoisena.
- RNN soveltuu tehtäviin, joissa syötteellä on selkeä järjestys ja jokainen tapahtuma riippuu aiemmasta, esimerkiksi luonnollisen kielen käsittely (esim. tunneanalyysi) tai aikasarjadata, esimerkiksi osakekurssien tai sään ennustaminen)

Vanishing gradient

- Esiintyy neuroverkkojen opetuksessa, erityisesti kun malli yrittää löytää pitkäaikaisia riippuvuuksia aikasarjadatasta
- Gradientti kertoo kuinka paljon painokertoimia päivitetään oppimisen aikana
- Kun painokertoimia päivitetään edestakaisin, on vaarana, että syvimmät kerrokset saavat hyvin pienen gradientin, mikä vaikeuttaa niiden painokertoimien päivittämistä, mikä taas johtaa siihen, että syvät kerrokset eivät opi hyödyllisiä ominaisuuksia tai eivät opi pitkiä riippuvuuksia

RNN vs LSTM (Long Short-Term Memory)

- RNN, jokainen yksikkö saa syötteen ja piilotilan (hidden state) edellisestä sekvenssistä tai aikasarjasta
- ”Vanishing gradient” eli katoavan gradientin ongelma, joka vaikeuttaa kauempana olevien riippuvuuksien oppimista
- Soveltuu paremmin lyhytaikaisiin riippuvuuksiin
- LSTM (Long Short-Term Memory) on suunniteltu ottamaan paremmin huomioon katoavan gradientin ongelma
- LSTM muistaa pidempiä sekvenssejä, jolloin kauempana olevat riippuvuudet voidaan ottaa paremmin huomioon

Tekstin käsittely mallinnusta varten

- Aineisto jaetaan pienemmiksi merkityksellisiksi yksiköiksi(token) (huom! Isot ja pienet kirjaimet, merkit ja pysäytyssanat)
- Kootaan tietty sanasto aineistosta.
- Jokainen yksikkö muutetaan vektoriksi joko one-hot-encoding- tai embedding- (sanan upotus) tekniikan avulla

Tokenizer – merkitysyksiköistä indekseihin

- Tokenizer-funktio muuttaa tekstin merkkijonoiksi. Oletuksena jaottelu tapahtuu sanatasolla, kaikki kirjaimet muutetaan pieniksi ja erikoismerkit suodatetaan
- Sanastolle valitaan sanamäärä
- fit_on_texts luo sanaston ja liittää jokaiseen sanaan yksilöllisen indeksin
- texts_to_sequences muuttaa alkuperäisen tekstin sekvensseiksi sanojen indeksien perusteella

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer(num_words = 5000)
tokenizer.fit_on_texts(samples)
sequences = tokenizer.texts_to_sequences(samples)
data = pad_sequences(sequences, maxlen = 100)
```

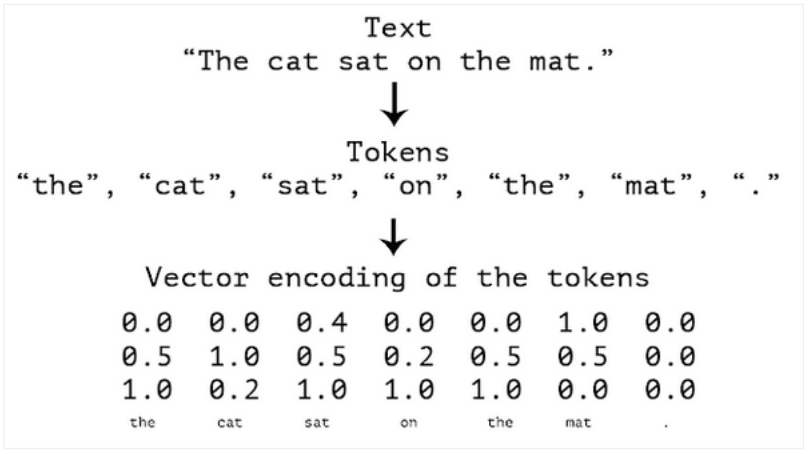


Figure 1 From text to tokens to vectors

https://freecontent.manning.com/deep-learning-for-text/

One-hot-encoding vs embedding

- One-hot-encoding-metodilla prosessoiduilla sanoilla ei ole tietoa sanojen välisestä suhteesta
- Prosessointi voi viedä aikaa varsinkin, jos sanoja on paljon
- Sanan upotuksessa sanat liitetään pienen ulottuvuuden vektoriin
- Sanat sijaitsevat lähellä toisiaan upotusavaruudessa
- Embedding-menetelmässä sanan upotuksessa sanavektorit saadaan erillisen oppimisalgoritmin avulla, joko erillisenä metodina tai tarkasteltavan mallin yhteydessä
- Vektorit upotusavaruudessa ovat koulutettavia muuttujia (trainable parametres), jotka muuttuvat oppimisen aikana

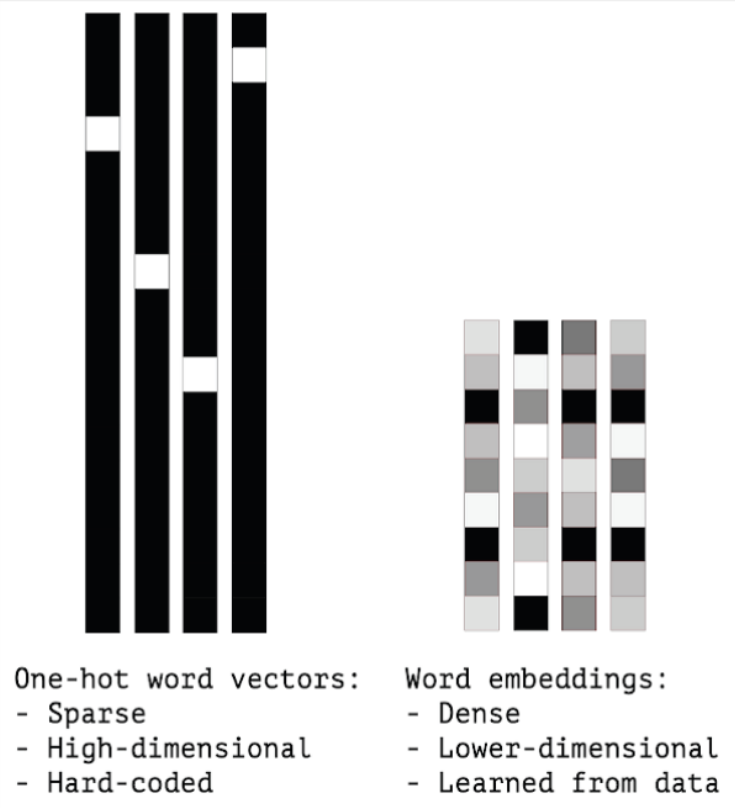


Figure 2 Although word representations obtained from one-hot encoding or hashing are sparse, high-dimensional, and hard-coded, word embeddings are dense, relatively low-dimensional, and learned from data.

https://freecontent.manning.com/deep-learning-for-text/

Upotus (embedding)

- Yleistä tekstin käsittelyssä
- Sanat ja merkkijonot muutetaan vektoreiksi

python Copy code

```python
from keras.layers import Embedding

# Määritä sanaston koko ja upotuksen ulottuvuus
vocab_size = 10000
embedding_dim = 50

# Luo Embedding-kerros
embedding_layer = Embedding(input_dim=vocab_size, output_dim=embedding_dim,
```

Embedding (upotus)-kerros mallin yhteydessä

- Jos Embedding-kerros laitetaan mallin ensimmäiseksi kerrokseksi, se saa syötteekseen kaksiulotteiseisen indeksitaulukon kokonaisluvuilla (otos ja sekvenssin pituus) ja antaa tuloksena kolmiulotteisen liukulukuisen sanavektoritensorin (otos, sekvenssin pituus, ja upotuksen ulottuvuus)
- Opetuksen edetessä sanavektori muovautuu muiden parametrien kanssa

```
from keras import models, layers

model = models.Sequential()
model.add(layers.Embedding(10000, 16, input_length = maxlen))
model.add(layers.Flatten())
model.add(layers.Dense(1, activation = 'sigmoid'))
```

Mallin syötteenä on 10 000 sanaa kokonaislukulistoina. Jokaisen listan pituus on määritelty (tässä vain muuttuja maxlen, mutta voi olla esimerkiksi 100 tai 200).
Malli muuttaa sanat 16-ulottuvuuden sanavektoreiksi

Flatten tasoittaa syötteen ulottuvuudet yhdeksi ulottuvuudeksi, koska sitä seuraa kerros, joka tarvitse yksiulotteiden syötteen.

Keras.Sequential()

- Keras, neuroverkkokirjasto
- Sequential(), pinoaa eri kerrokset päällekkäin
- Yksi input tensor ja yksi output tensor
  - Tensori yleistää käsitteet skalaarista (esim. 5), vektorista (esim. [1,2,3], matriisista (esim. [[1,2], [3,4]]) korkeampiin ulottuvuuksiin, kuten värivalokuva (korkeus, leveys, väri, RGB)
- https://fi.wikipedia.org/wiki/Tensori
- https://keras.io/guides/sequential_model/

Dense layer

- Neuroverkkomallin kerros, jossa jokainen neuroni on yhdistetty edellisen kerroksen jokaiseen neuroniin
- KerasLayer (syötetty data)
- Dense layer : 16 neuronia ja relu-aktivaatio (piilokerros)
- Dense_1 : on 1 neuroni, malli tuottaa yhden lopputuloksen eli binaarisen 0 tai 1 (tuloste)

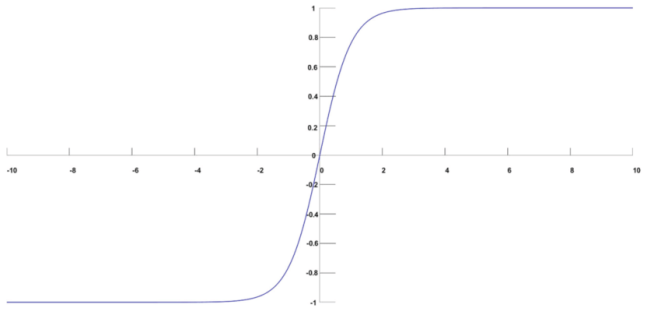
```
model = tf.keras.Sequential()
model.add(hub_layer)
model.add(tf.keras.layers.Dense(16, activation='relu'))
model.add(tf.keras.layers.Dense(1))

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer (KerasLayer)    (None, 50)                48190600

 dense (Dense)               (None, 16)                816

 dense_1 (Dense)             (None, 1)                 17

=================================================================
Total params: 48191433 (183.84 MB)
Trainable params: 48191433 (183.84 MB)
Non-trainable params: 0 (0.00 Byte)
```

Harha (Bias)

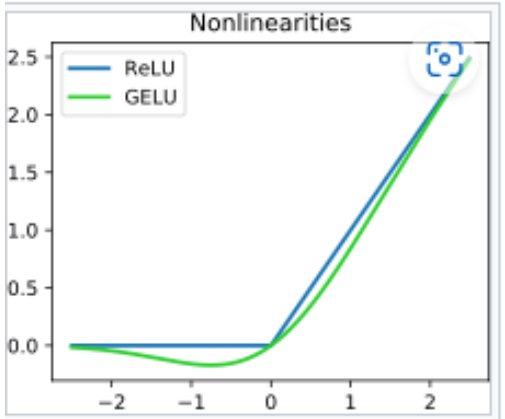
- On yksi syötteistä, jonka tarkoituksena on varmistaa, että luokitus saadaan reagoimaan painokertoimien muutoksiin
- Kiinteä vakio, esimerkiksi 1.
- Asettaa aktivaation tason, joka ei ole suoraan riippuvainen syötteestä

Aktivaatio-funktio (1)



- **Rectified Linear Unit (ReLU)**
- Jos syöte on positiivinen, palauttaa syötteen arvon, jos negatiivinen palauttaa nollan
- Auttaa vähentämään gradienttien häviämistä (vanishing gradient problem) toisin kuin esim sigmoid
- Käytetään usein piilokerroksessa
- https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

- **Softmax**
- Moniluokkaisen luokittelun lopussa
- https://en.wikipedia.org/wiki/Softmax_function

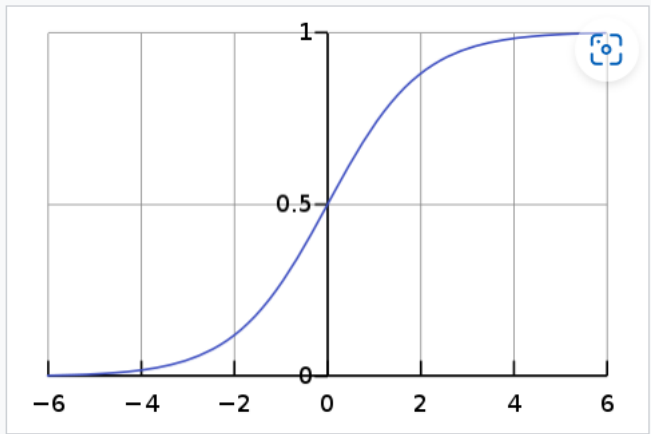


Plot of the ReLU rectifier (blue) and GELU (green) functions near $x = 0$

IoT
TAITOTALO

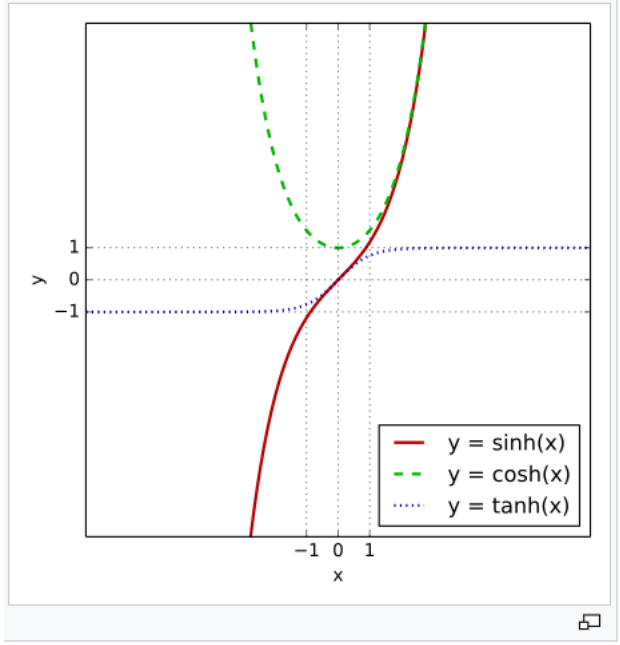
Aktivaatio-funktio (2)

- **Sigmoidfunktio**
- Binaariset luokittelutehtävät
- https://en.wikipedia.org/wiki/Sigmoid_function



The logistic curve

- **Hyperbolinen tangenttifunktio**
- Arvot vaihtelevat välillä -1, 1
- https://en.wikipedia.org/wiki/Hyperbolic_functions



IoT
TAITOTALO

Aktivaatio-funktio (3)

- Rikkoo mallin lineaarisuutta, ilman sitä malli koostuisi lineaarisista transformaatioista
- Määrittää kuinka neuronin lähtöarvo lasketaan syötearvojen perusteella
- Siirtää gradiantteja taaksepäin backpropagation-vaiheessa

Painokertoimet (weights) ja Backpropagation

- Painokertoimet saavat alussa usein satunnaisen arvon
- Mallin opetuksen aikana malli tekee ennustuksia ja vertaa niitä todellisiin arvoihin
- Backpropagation tarkoittaa painokertoimien muutoksen valuttamista aiemmille tasoille
- Varsinaisena tavoitteena on virheiden minimoiminen ennusteiden ja todellisten arvojen välillä
- Laskee gradientin sille kuinka paljon virhe muuttuu jokaisen painon suhteen

Optimointi (optimizer)

- Miten mallit painokertoimet päivitetään koulutuksen aikana, jotta voisi minimoida tappion (loss function) eli että mallin tekemät virheet pienenevät mallia kouluttaessa
- *Adam* on hyvä oletusfunktio, koska se toimii eri malleissa ilman erityisempää hyperparametrien säätämistä

```
model.compile(optimizer='adam',
              loss=tf.losses.BinaryCrossentropy(from_logits=True),
              metrics=[tf.metrics.BinaryAccuracy(threshold=0.0, name='accuracy')])
```

Loss funktio (mittari)

- Mittaa kuinka hyvin malli onnistuu ennustamaan todelliset arvot ja/tai luokat
- Binary Cross-Entropy Loss käytetään binäärinen luokittelutehtävä
- Muita esim. Mean Squared Error (MSE), jos regressiotehtävä
- Mittareihin voi lisätä myös muita mittareita kuten ’accuracy’ eli tarkkuus

Validointi-aineisto

- Erillinen aineisto tulosten mallin validointiin opetus- ja testiaineiston lisäksi
- Näin pyritään pitämään testiaineisto pois mallin kouluttamisesta ja näin estämään mallin ylioppimista

Kouluttaminen

- Epochs = kierrokset
- Batch_size = yhden erän koko
- Verbose 1 = tulostaa kouluttamisen tulokset kierros kerrallaan (jos 0 ei tulostusta)

- Arvoja on mahdollista muuttaa
- Epoch ja batch size vaikuttavat nopeuteen ja vaadittavaan muistiin opetuksessa

```
[7]  history = model.fit(partial_x_train,
                         partial_y_train,
                         epochs=40,
                         batch_size=512,
                         validation_data=(x_val, y_val),
                         verbose=1)
```

Mikä on hyvä batch size

- What is a good batch size for RNN?

This is because the batch size needs to fit the memory requirements of the GPU and the architecture of the CPU. So, the acceptable values for the batch size are 16, 32, 64, 128, 256, 512 and 1024!

https://medium.com/data-science-365/determining-the-right-batch-size-for-a-neural-network-to-get-better-and-faster-results-7a8662830f15

Mallin arviointi

- Millä tavalla mallia voisi parantaa?

```
[8]  results = model.evaluate(test_examples, test_labels)

     print(results)
```

```
782/782 [==============================] - 132s 168ms/step - loss: 0.5854 - accuracy: 0.8464
[0.5853767991065979, 0.8464000225067139]
```

This fairly naive approach achieves an accuracy of about 87%. With more advanced approaches, the model should get closer to 95%.

Ennustaminen

Viidestä ensimmäisestä rivistä 4/5 osui oikeaan

```
[15] predictions = model.predict(test_examples)

     782/782 [==============================] - 126s 160ms/step
```

```
[22] print("Test examples:")
     print(test_examples[:5])

     Test examples:
     [b"There are films that make careers. For George Romero, it was NIGH
      b"A blackly comic tale of a down-trodden priest, Nazarin showcases
      b'Scary Movie 1-4, Epic Movie, Date Movie, Meet the Spartans, Not a
      b'Poor Shirley MacLaine tries hard to lend some gravitas to this ma
      b'As a former Erasmus student I enjoyed this film very much. It was
```

```
[23] print(test_labels[:5])

     [1 1 0 0 1]
```

```
print("Predictions:\n")
print(predictions[:5])  # Tulosta ensimmäiset 5 ennustetta

Predictions:

[[-3.5601218]
 [ 3.7728593]
 [-4.592485 ]
 [-8.700286 ]
 [ 9.175387 ]]
```

IoT TAITOTALO

Linkit

https://www.manning.com/books/deep-learning-with-python

https://freecontent.manning.com/deep-learning-for-text/

Visualisointi:

https://www.kaggle.com/code/ashwatidinesh/imdb-dataset-eda

Opinnäytetyö:

https://www.theseus.fi/bitstream/handle/10024/751576/Kaasikoja_Kaspar.pdf?sequence=2

Tehtävä 1

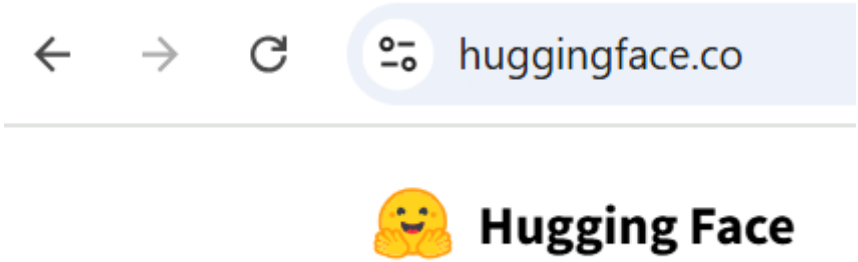
- Kokeile saada mallin tarkkuus 95 prosenttiin muuttamalla eri parametreja ja/tai funktioita

tf2_text_classification.ipynb - Colaboratory (google.com)

- Selvitä mitä muita mittareita voi käyttää tarkkuuden lisäksi (accuracy)?

Tehtävä 2 drug review, sentiment-analyysi

1. Tutustu alkuperäiseen artikkeliin: https://kdd.cs.ksu.edu/Publications/Student/kallumadi2018aspect.pdf
2. Tutustu notebookin: https://www.kaggle.com/code/samuelrsnen/case-3-patient-drug-review-final/notebook (Kagglessa myös muita vastaavia)
3. Tutustu datasta tehtyihin visualisointeihin: https://www.kaggle.com/code/harshjain123/drugs-review-sentiment/notebook
4. Kerro omin sanoin kolme tärkeintä asiaa mitä opit jokaisesta tiedostosta. Voit lisätä myös kuvakaappauksia. Perustele vastauksesi.
5. Voiko lääkkeen arvosanan ennustaa kirjallisesta arviosta tehdyn analyysin perusteella?
6. Laske sekaannusmatriisi ja Cohen’s kappa elokuva-tiedostolle (Text Classification with Movie Reviews). Kuvakaappaukset mukaan
7. Extra: jos aikaa jää, rakenna oma analyysityökalu lääkkeiden arvosanan ennustamiseen


huggingface.co
Hugging Face