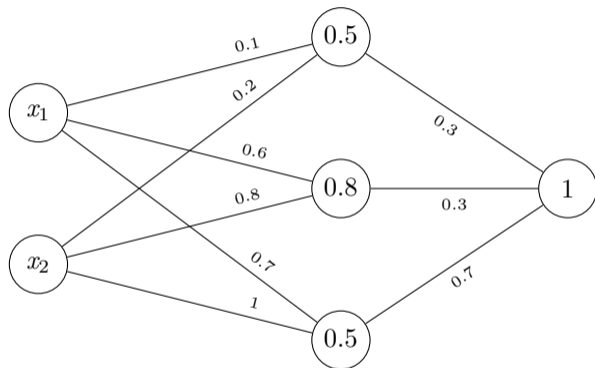


## Recap



$x_1$	$x_2$	$y$
0	0	0.86169636
1	0	0.87786007
1	1	0.891605
10	10	0.90814614
$\vdots$	$\vdots$	$\vdots$

Figure: Neural network with randomly initialized weights

# Neural Networks (2/2)

## Sprachverarbeitung (VL + Ü)

Nils Reiter

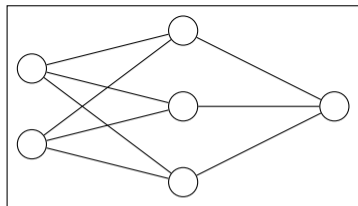
July 11, 2023

# Dimensions

- ▶ Dimensionality of neural networks major source of confusion

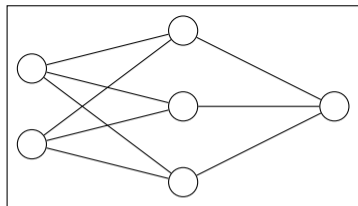
## Dimensions

- ▶ Dimensionality of neural networks major source of confusion
- ▶ In this example •
  - ▶ Single input object represented with two numbers (= 1D)
  - ▶ Output is a single number
- ▶ Dimensions:
  - ▶ Input data set: 2D (because multiple instances)
  - ▶ Output data: 1D (a list of single numbers)




## Dimensions

- ▶ Dimensionality of neural networks major source of confusion
- ▶ In this example •
  - ▶ Single input object represented with two numbers (= 1D)
  - ▶ Output is a single number
- ▶ Dimensions:
  - ▶ Input data set: 2D (because multiple instances)
  - ▶ Output data: 1D (a list of single numbers)




```
# ...  
model = keras.Sequential()  
model.add(layers.Input(shape=(num_numbers,)))  
# ...
```

# Shape

- ▶ Description of the dimensionality of the data
- ▶ A vector of numbers, giving the number of elements for each dimension
- ▶ Python tuple
  - ▶ List with fixed length: `x = (5,3,1)` #a tuple
  - ▶  Tuple with one element printed as `(5,)` or `5`

# Shape

- ▶ Description of the dimensionality of the data
- ▶ A vector of numbers, giving the number of elements for each dimension
- ▶ Python tuple
  - ▶ List with fixed length: `x = (5,3,1)` #a tuple
  - ▶  Tuple with one element printed as `(5,)` or `5`

```
x = np.zeros(5) # array([0., 0., 0., 0., 0.])
x.shape # returns (5,)
x = np.zeros((3,5))
# array([[0., 0., 0., 0., 0.],      # a 3x5 matrix
#        [0., 0., 0., 0., 0.],
#        [0., 0., 0., 0., 0.]])
x.shape # returns (3,5)
```

# Input Shape for a Neural Network

```
model.add(layers.Input(shape=(num_numbers,)))
```

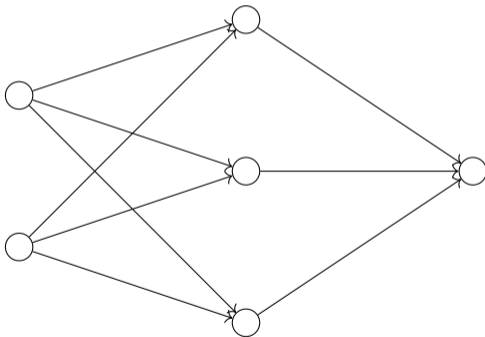
- ▶ We can specify an input shape with more dimensions!



# Input Shape for a Neural Network

1D

```
model.add(layers.Input(shape=(2,))) # input is vector with two elements
```



# Input Shape for a Neural Network

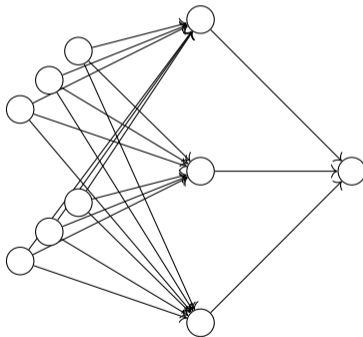
2D

```
model.add(layers.Input(shape=(2,3))) # input is 2x3 matrix
```

# Input Shape for a Neural Network

2D

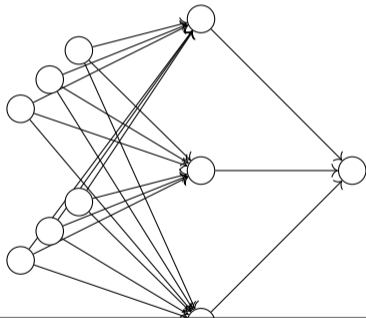
```
model.add(layers.Input(shape=(2,3))) # input is 2x3 matrix
```



# Input Shape for a Neural Network

2D

```
model.add(layers.Input(shape=(2,3))) # input is 2x3 matrix
```

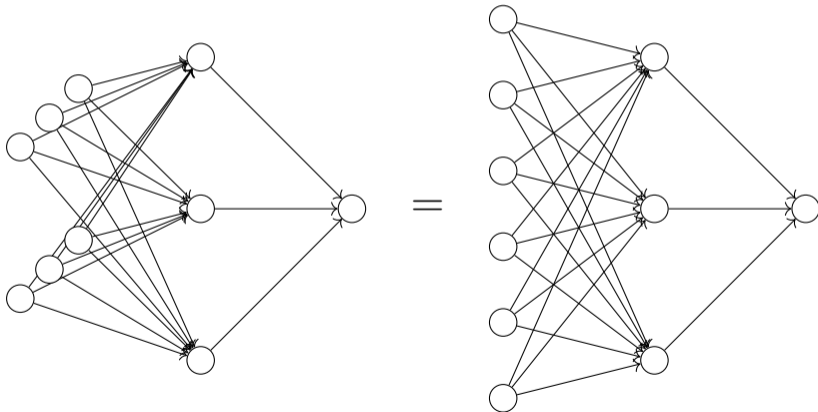


$$a(w_{1,1,3}x_{1,1} + w_{1,2,3}x_{1,2} + w_{1,3,3}x_{1,3} + w_{2,1,3}x_{2,1} + w_{2,2,3}x_{2,2} + w_{2,3,3}x_{2,3} + b_{2,3})$$

# Input Shape for a Neural Network

2D

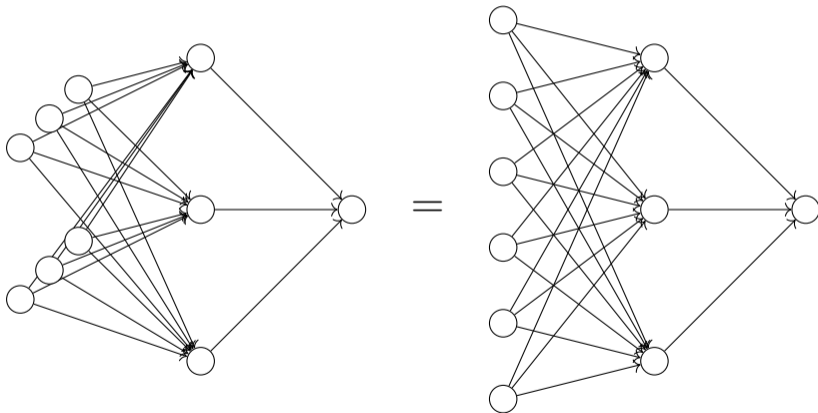
```
model.add(layers.Input(shape=(2,3))) # input is 2x3 matrix
```



# Input Shape for a Neural Network

2D

```
model.add(layers.Input(shape=(2,3))) # input is 2x3 matrix
```



➔ Input shapes can be ›flattened‹, if next layer fully connected

Section 1

Overfitting

# Introduction

- ▶ »Fitting«: Train a model on data (= »fit« it to the data)
  - ▶ Underfitting: The model is not well fitted to the data, i.e., accuracy is low
  - ▶ Overfitting: The model is fitted too well to the data, i.e., accuracy is high



## Introduction

- ▶ »Fitting«: Train a model on data (= »fit« it to the data)
  - ▶ Underfitting: The model is not well fitted to the data, i.e., accuracy is low
  - ▶ Overfitting: The model is fitted too well to the data, i.e., accuracy is high

### Why is overfitting a problem?

# Introduction

- ▶ »Fitting«: Train a model on data (= »fit« it to the data)
  - ▶ Underfitting: The model is not well fitted to the data, i.e., accuracy is low
  - ▶ Overfitting: The model is fitted too well to the data, i.e., accuracy is high

## Why is overfitting a problem?

- ▶ We want the model to behave well »in the wild«
- ▶ It needs to generalize from training data
  - ▶ And, ideally, from our test data (!)
- ▶ If it is overfitted, it works very well on »lab data«, and very badly in the real world

# Intuition

≈ Learning by heart

## Example

- ▶ Learning by heart gets you through the test
  - ▶ I.e., systems achieve high performance

# Intuition

≈ Learning by heart

## Example

- ▶ Learning by heart gets you through the test
  - ▶ I.e., systems achieve high performance
- ▶ You are unable to apply your knowledge to situations not exactly as in the test
  - ▶ I.e., system performance is lower in the wild

# Intuition

≈ Learning by heart

## Example

- ▶ Learning by heart gets you through the test
  - ▶ I.e., systems achieve high performance
- ▶ You are unable to apply your knowledge to situations not exactly as in the test
  - ▶ I.e., system performance is lower in the wild

Wie schätzen Sie die Situation ein?

Die Fußgängerin kann unachtsam die Fahrbahn betreten

Ich kann unvermindert weiterfahren

Der Fußgänger mit dem Mofa kann plötzlich die Richtung ändern



## Real-World Examples

- ▶ ML/DL to detect COVID-19 from CT scans of the lung
  - ▶ Article for general public
- ▶ Collection of examples for overfitting

Roberts et al. (2021)  
MIT Technology Review  
[stackexchange.com](https://www.stackexchange.com)


# Overfitting and Neural Networks

## Classical machine learning

- ▶ Feature selection can avoid relying on irrelevant features
- ⚠ Only one source for overfitting

# Overfitting and Neural Networks

## Classical machine learning

- ▶ Feature selection can avoid relying on irrelevant features
- ▶  Only one source for overfitting

## Neural networks are overfitting machines

- ▶ Layered architecture  $\Rightarrow$  Any relation between  $x$  and  $y$  can be learned
  - ▶ including a fixed set of if/else rules

## Techniques against overfitting

- ▶ Regularization (not today)
- ▶ Dropout



## Subsection 1

### Dropout

# Dropout

- ▶ Regularization: Numerically combatting overfitting
- ▶ Dropout: Structurally combatting overfitting

Hinton et al. (2012)

# Dropout

- ▶ Regularization: Numerically combatting overfitting
- ▶ Dropout: Structurally combatting overfitting
  - ▶ A new hyperparameter  $\pi = [0; 1]$
  - ▶ In each epoch, every weight is set to zero with a probability of  $\pi$

Hinton et al. (2012)

# Dropout

## Example

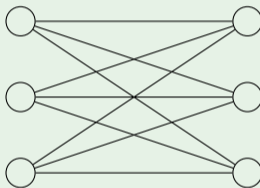


Figure: Dropout  $\pi = 0.5$ , visualized

# Dropout

## Example

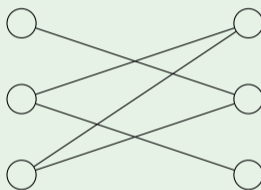


Figure: Dropout  $\pi = 0.5$ , visualized, Epoch 0

# Dropout

## Example

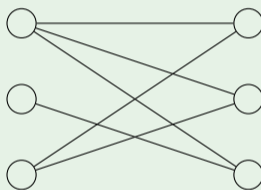


Figure: Dropout  $\pi = 0.5$ , visualized, Epoch 1

# Dropout

## Example

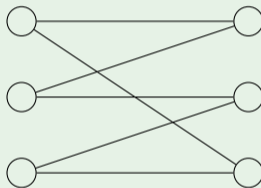


Figure: Dropout  $\pi = 0.5$ , visualized, Epoch 2

# Dropout

## Implementation

- ▶ Why?
  - ▶ Dropout forces the network to learn redundancies



# Dropout

## Implementation

- ▶ Why?
  - ▶ Dropout forces the network to learn redundancies
- ▶ How?
  - ▶ In Keras, dropout is realized as additional layer
  - ▶ Applies to the layer before the dropout layer

```
1 model.add(layers.Dense(10)) # no edges dropped
2 model.add(layers.Dense(20)) # edges are dropped here
3 model.add(layers.Dropout(0.5))
```

## Section 2

### Closing and Looking Ahead

## Checkliste zu NLP-Experimenten zur Klassifikation

Stand: 20. Dezember 2022

### Hinweise

- Wenn Ihr Task kein Klassifikationstask ist, dann ist dieser Fragebogen nicht für Sie.
  - Keine Klassifikationstasks sind z.B. Generierung oder Übersetzung.
- Markieren Sie alle Punkte die Sie planen umzusetzen.
- Machen Sie einen Termin in der Sprechstunde, wenn Ihnen Punkte unklar sind.
- Der Fragebogen ist keine Prüfung, sondern dient als Hilfestellung, bei der Experimentplanung an alles zu denken, auf Ideen zu kommen und ggf. die richtigen Fragen zu stellen. Diese klären wir dann idealerweise im Gespräch.
- Der Fragebogen repräsentiert eine Planung. Gewisse Abweichungen von der Planung sind normal und zu erwarten.
- Bei Fragen schreiben Sie gerne eine E-Mail an [nila.reiter@uni-koeln.de](mailto:nila.reiter@uni-koeln.de) oder melden Sie sich gerne zu einer Sprechstunde an. Wenn Sie rein technische Fragen zur Infrastruktur haben, schreiben Sie bitte an [spinfo-admin@uni-koeln.de](mailto:spinfo-admin@uni-koeln.de).

### Der Task

1. Die Aufgabe heißt: \_\_\_\_\_
2. Es handelt sich um  Textklassifikation,  Sequence Labeling, oder  Sonstiges: \_\_\_\_\_
3. Die zu klassifizierenden Instanzen sind: \_\_\_\_\_
4. Es gibt \_\_\_\_\_ Kategorien/Klassen.
5. Einer Instanz kann  genau eine oder  mehrere Klassen zugewiesen werden.

### Die Daten

1. Annotierte Daten  liegen bereits vor oder  müssen noch erstellt werden.
2. In den Daten sind \_\_\_\_\_ Instanzen (von o.g. Typ) annotiert.
3. Die Klassen sind
  - gleichverteilt (d.h. jede Klasse ist ungefähr gleich häufig)
  - unterschiedlich verteilt, und zwar: \_\_\_\_\_

### Die Annotationen

*Nur relevant, wenn neue Daten annotiert werden sollen. (Frage Task.1)*

1. Annotationsrichtlinien
  - Ich verwende die folgenden, bereits existierenden Annotationsrichtlinien: \_\_\_\_\_
    - Mit diesen wurde ein Inter-Annotator-Agreement von \_\_\_\_\_ erzielt (Metrik: \_\_\_\_\_).
  - Ich schreibe neue Annotationsrichtlinien.

2. Annotator:innen
  - Ich annotiere selbst.
  - Ich rekrutiere Annotator:innen aus meinem Freundes-/Bekannteskreis.
  - Ich sammle Annotationen über eine Umfrage, z.B. mittels LimeSurvey.
  - Ich sammle Annotationen über crowd sourcing.
3. Annotationsworkflow
  - Annotator:innen treffen eine Annotationsentscheidung auf der Basis eines Kontextes von \_\_\_\_\_ Wörtern, \_\_\_\_\_ Sätzen, \_\_\_\_\_ Zeilen, \_\_\_\_\_ Absätzen, \_\_\_\_\_ \_\_\_\_\_; oder  sie verwenden den gesamten Text als Kontext.
  - Sie können dabei außerdem die folgenden Wissensquellen verwenden:  Wikipedia,  Lexika,  Wörterbücher
4. Anforderungen an Annotationssoftware
  - Annotator:innen müssen Spalten selbst markieren können.
  - Annotator:innen müssen neue Kategorien oder Labels ergänzen können.

### Die Baseline

- Weil die Klassen ungleich verteilt sind, bietet sich eine majority baseline an. Diese erzielt eine Accuracy von \_\_\_\_\_ %.
- Weil die Klassen gleich verteilt sind, bietet sich eine random baseline an. Diese erzielt eine Accuracy von \_\_\_\_\_ %.
- Eine weitere mögliche Baseline ist: \_\_\_\_\_  
Diese erzielt eine Accuracy von \_\_\_\_\_ %.
- Eine weitere mögliche Baseline ist: \_\_\_\_\_  
Diese erzielt eine Accuracy von \_\_\_\_\_ %.

### Das Experiment

1. Ich möchte das folgende oder die folgenden Verfahren verwenden:
  - Entscheidungsbaum / Decision Tree (DT)
  - Naive Bayes
  - Support Vector Machines (SVM)
  - Logistic Regression
  - Neural Networks (NN)
    - Feed-Forward Neural Networks
    - Convolutional Neural Networks
    - Recurrent Neural Networks
    - Transformer-Architektur (BERT & co.)
  - Sonstige: \_\_\_\_\_
2. Ich möchte die folgenden Features verwenden
  - Metadaten: \_\_\_\_\_  
\_\_\_\_\_
  - Inhaltsdaten, z.B. aus Texten:
    - Worthäufigkeiten (von allen Wörtern), auch bekannt als bag of words

- Häufigkeiten von Wörtern aus folgenden Wortlisten: \_\_\_\_\_
  - Embeddings (z.B. Word Embeddings)
  - Sequenzielle Information (d.h. Klassifikationsergebnisse für Elemente davor oder danach)
  - $N$ -Gram-Häufigkeiten, mit  $N \leq$  \_\_\_\_\_
  - Thematische Informationen aus einem Topic Model (z.B. Latent Dirichlet Allocation, LDA)
3. Meine Features haben die folgenden Datentypen:
- Numerisch: \_\_\_\_\_ (Anzahl Features)
  - Kategorial: \_\_\_\_\_ (Anzahl Features)
4. Testdaten
- Ich teile meinen o.g. Datensatz selbst in Trainings- und Testdaten auf. \_\_\_\_\_% der Instanzen werden als Trainingsdaten verwendet.
  - Ich verwende  $N$ -fold cross validation, mit  $N =$  \_\_\_\_\_.
  - Trainings- und Testdaten sind bereits aufgeteilt, z.B. weil es Daten aus einem shared task sind.
5. Ich variere und vergleiche
- die Größe des Trainingsdatensatzes (z.B. 100, 1000, 10000 Instanzen für den Trainingsdatensatz)
  - die Menge an oder Art von Features die verwendet werden (z.B. inhaltliche vs. sprachliche Features)
  - das Verfahren als solches oder Parameter davon (z.B. NN vs. SVM)
  - die Vorverarbeitung (z.B. Groß- und Kleinschreibung)
6. Meine Hypothese ist: \_\_\_\_\_
- \_\_\_\_\_
- \_\_\_\_\_
- \_\_\_\_\_

### Die Auswertung und Evaluation

1. Ich verwende die Evaluationsmetrik(en)
- Accuracy
  - Precision
  - Recall
  - F-Messure
  - Area under curve (AUC)
  - Sonstige: \_\_\_\_\_
- Meine Testdaten sind stark unbalanciert (Frage Daten.3), daher verwende ich die Metriken in der Mikro- und Makro-Average-Variante.
2. Für meine Fehleranalyse inspiziere ich \_\_\_\_\_ Instanzen manuell.

### Die praktische Umsetzung

1. Ich verwende die Programmiersprache
- Python
  - Java
  - R
  - \_\_\_\_\_
2. Hardware-Ausstattung und Vorkenntnisse
- Ich verfüge über einen Computer
    - der auch mal über Nacht durchlaufen kann, wenn eine Berechnung etwas länger dauert.
    - der eine GPU mit CUDA-Unterstützung hat oder ein Mac mit M1/M2-Prozessor ist.
    - der ausreichend freien Plattenspeicher hat.
  - Ich möchte Berechnungen auf einem Server der Universität laufen lassen.
    - Ich kann mich per SSH auf einem Server einloggen.
    - Ich weiß wie ich auf einer Kommandozeile ein Programm laufen lasse.

## Future Paths for Computational Linguistics

*Large language models have outperformed our ability to evaluate them.*

(Mark Finlayson; words to that effect)

# Future Paths for Computational Linguistics

*Large language models have outperformed our ability to evaluate them.*

(Mark Finlayson; words to that effect) Future areas

- ▶ What can LLMs do and what not?
- ▶ Better evaluation may include consumption of resources
- ▶ Multimodality: Text and music, text and images, ...
- ▶ More high-level applications
- ▶ Scientific use  $\neq$  general use
  - ▶ More control needed for scientific use



Enjoy the Summer!