Übungsblatt 8 - Musterlösung

Aufgabe 8.1

a) Was ist ein neuronales Netz?
Neuronale Netze sind eine beliebte Machine Learning-Methode angelehnt an die Funktion eines Gehirns. Sie lernen durch Feedback und Anpassung der Gewichte in der Netzstruktur.

Was versteht man unter der Aktivierungsfunktion?
Sie bestimmt in Abhängigkeit von den Eingabewerten, welche Informationen an die folgenden Neuronen weitergeleitet werden sollen bzw. ob das Neuron aktiv ist und feuert oder nicht.

Was versteht man unter Input Layer, Output Layer und Hidden Layer?
Die Input Layer besteht aus Knoten, die den Input aufnehmen, bei Bilderkennung z.B. die Graustufe eines Pixels. Die Output Layer besteht aus Knoten, die den Output des Netzwerks wiedergeben, bei der Bilderkennung z.B. die Klassifizierung des Bildinhalts (Hund, Katze, Mensch, ...). Die Hidden Layers fungieren als Übermittler zischen Input und Output Layer und dienen der Mustererkennung. Je komplexer die Aufgabe, desto mehr Hidden Layers werden benötigt.

b) Ein neuronales Netz besteht aus vielen verstellbaren Parametern, die in einen (zufälligen) Anfangszustand versetzt werden. Sogenannte Trainingsdaten beinhalten die Lösung der Aufgabe bereits, sodass der Output des Netzes damit abgeglichen werden kann. Die Parameter im Netz werden so lange angepasst bis der Output des Netzes und die erwarteten Ergebnisse in großen Teilen übereinstimmen (=Backpropagation).

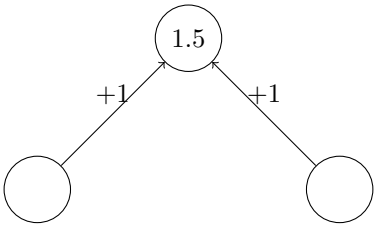
c) R steht für Recurrent, d.h. das RNN kann beliebig langen Input verarbeiten (z.B. einen Satz). Dadurch ist es geeignet für das Erfassen von Sprache und Kontexten.

d) LSTMs regulieren die Gewichte zusätzlich, sodass der Gradient stabil bleibt und weniger Probleme entstehen. Dadurch können sie besonders gut weitreichende Dependenzen erfassen.

e) Embeddings können als Vektorrepräsentationen von Wörtern verstanden werden. Zum einen kann man damit semantische Ähnlichkeiten untersuchen und z.B. Wörter in einem multidimensionalen Raum clustern und zum anderen sind Embeddings eine sinnvolle Vorstufe für viele NLP-Problemstellungen wie POS-Tagging, sentiment analysis etc.

Aufgabe 8.2

1)



Wir modellieren das logische Und. Nehmen wir an, dass wir zwei atomare Ausdrücke betrachten, die wie unsere Inputknoten benannt sind. Uns liegt also der logische Ausdruck **I1** $\wedge$ **I2** vor. I1 und I2 können jeweils den Wert 0 oder 1 haben, also ergeben sich vier Kombinationsmöglichkeiten:

I1 ist wahr und I2 ist wahr
I1 ist wahr und I2 ist falsch
I1 ist falsch und I2 ist wahr
I1 ist falsch und I2 ist falsch

Der Gesamtausdruck **I1** $\wedge$ **I2** wertet nur dann zu wahr aus, wenn sowohl I1 als auch I2 wahr ist. Ansonsten wertet er zu falsch aus. Das heißt, dass die Ausgabe von Knoten O1 in einem Fall 1 ist und in den anderen Fällen 0.

Um die Eingabe an Knoten O1 zu berechnen, multiplizieren wir erst die Aktivierung von I1 (die entweder 0 oder 1 beträgt) mit dem entsprechenden Gewicht auf der von I1 ausgehenden Kante. Dann machen wir das gleiche bei I2 und summieren die beiden erhaltenen Produkte.

Eingabe an Knoten I1	Eingabe an Knoten I2	Eingabe an Knoten O1	Ausgabe von Knoten O1
0	0	(0*1+0*1=) 0	0
1	0	(1*1+0*1=) 1	0
0	1	(0*1+1*1=) 1	0
1	1	(1*1+1*1=) 2	1

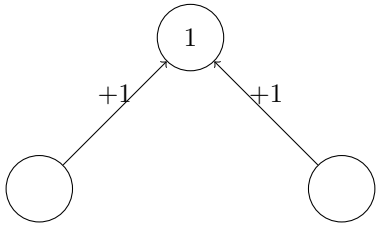
2)

Ein Ausdruck mit dem logischen Oder wertet immer dann zu wahr aus, wenn mindestens einer der beiden Teilausdrücke zu wahr auswertet, also erhalten wir analog zum vorhergehenden Aufgabenteil für **I1** $\vee$ **I2** folgende Tabelle:

Eingabe an Knoten I1	Eingabe an Knoten I2	Ausgabe von Knoten O1
0	0	0
1	0	1
0	1	1
1	1	1

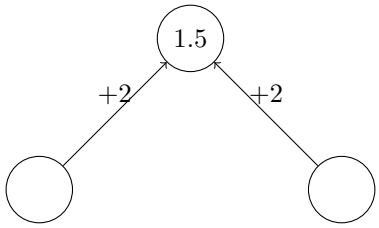
Wir müssen nun die Gewichte und Aktivierungsfunktion so bestimmen, dass die Eingabe an Knoten O1 zu dieser Ausgabe führt. Hier sind ein paar Beispiele:

a) Wir nehmen uns das Perzeptron für UND, aber setzen den Schwellenwert für die Aktivierung von Knoten O1 auf 1 herab.



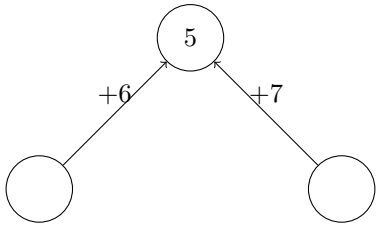
Eingabe an Knoten I1	Eingabe an Knoten I2	Eingabe an Knoten O1	Ausgabe von Knoten O1
0	0	(0*1+0*1=) 0	0
1	0	(1*1+0*1=) 1	1
0	1	(0*1+1*1=) 1	1
1	1	(1*1+1*1=) 2	1

b) Wir nehmen uns das Perzeptron für UND, aber setzen die Gewichte auf 2, sodass ein einziger feuernder Inputknoten ausreicht, um den Schwellenwert zu erreichen bzw. zu überschreiten.



Eingabe an Knoten I1	Eingabe an Knoten I2	Eingabe an Knoten O1	Ausgabe von Knoten O1
0	0	(0*2+0*2=) 0	0
1	0	(1*2+0*2=) 2	1
0	1	(0*2+1*2=) 2	1
1	1	(1*2+1*2=) 4	1

c) Wir setzen komplett andere Werte. Es gibt unendlich viele Wertekombinationen, mit denen sich das logische Oder modellieren ließe. Solange die Gewichte höher sind als der Schwellenwert, klassifiziert unser Perzeptron logische Oder-Ausdrücke korrekt.



Eingabe an Knoten I1	Eingabe an Knoten I2	Eingabe an Knoten O1	Ausgabe von Knoten O1
0	0	(0*6+0*7=) 0	0
1	0	(1*6+0*7=) 6	1
0	1	(0*6+1*7=) 7	1
1	1	(1*6+1*7=) 13	1