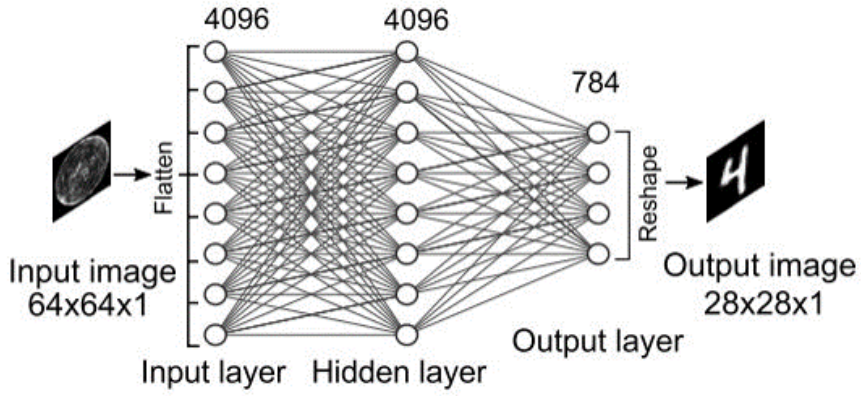

Quelle: Zhu, Changyan, et al. „Image reconstruction through a multimode fiber with a simple neural network architecture.“ Sci. Rep. 11.1 (2021): 1-10.

Praktikum Neuronale Netze in der Bildverarbeitung

Versuchsanleitung

Bildrekonstruktion handgeschriebener Ziffern bei einer Multimodefaser

Stefan Rothe, Tom Glosemeyer, Qian Zhang, Dennis Pohle, Jürgen Czarske

16. Dezember 2022

Inhaltsverzeichnis

1 Einleitung

Wie Denken wir? - Das menschliche Individuum ist nicht nur in der Lage, durch seine Sinnesorgane die Umgebung wahrzunehmen. Es kann auch mit dieser interagieren, beispielsweise durch Bewegung. Während bei der Wahrnehmung die Umgebungsgrößen in Reize umgewandelt werden, wandelt ein Muskel diese Reize in Bewegung um. Das „Denken“ bezeichnet in dieser Darstellung die Umwandlung und Verteilung der Reize zwischen sensorischer- und aktorischer Ebene durch ein komplexes Netzwerk aus Nerven [2].In der allgemeinen Auffassung wird die kleinste Einheit dieses Nervensystems als Neuron bezeichnet. Es wird angenommen, das dieses verschiedene Reize aufnehmen, gegeneinander wichten und an weitere Neuronen weitergeben kann. Stellt man sich die Reize als eine einfache Zahl vor, lässt sich der Sachverhalt leicht in einem mathematischen Modell darstellen und bildet die Grundlage für die Systeme künstlicher Neuronaler Netze. Diese und alle weiteren grundlegenden Betrachtungen können in [1] nachgelesen werden.

Ziel des zweiten Praktikumversuchs

Im zweiten Versuch wird mittels neuronaler Netze eine Bildrekonstruktion bei einer Multimodefaser durchgeführt. Neben der Einführung in die grundlegenden physiklischen Prinzipien der Multimodefaser, bilden die fundamentalen Techniken zur Konstruktion und zum Training neuronaler Netze aus dem ersten Versuch die Basis dieser Praktikumsaufgabe. Bei der Bildrekonstruktion werden Regressionen durchgeführt (Vgl. Klassifizierung aus Versuch 1). In diesem Fall sind sowohl das input, als auch das output layer vektorisierte Bilder (vgl. Label bei der Klassifizierung aus Versuch 1), die jeweils zur Ein- bzw. Ausgabe von Information dienen. Kern dieses Versuchs wird der Umgang mit Overfitting sein, welches ein häufig zu beobachtendes Problem beim Training neuronaler Netze ist. Overfitting charakterisiert sich durch die Divergenz der Lernkurven von Training und Validierung. Das bedeutet, dass das neuronale Netz zwar gut die Aufagben auf Basis bekannter Trainingsdaten lösen kann, es jedoch nicht schafft das vorliegende Problem zu generalisieren, um erfolgreich mit unbekannten Daten umzugehen. Im Rahmen dieses Praktikums werden Sie dem Overfitting begegnen und geeignete Maßnahmen (Data Augmentation) zur Vorbeugung, bzw. Überwindung von Overfitting kennenlernen. Schließlich lernen Sie neben dem Multilayer Perzeptron eine weitere, bekannte Netzarchitektur zur Bildverarbeitung kennen. Das U-Net verwendet Faltungsschichten, sog. *convolutional layer*, die Sie bereits im ersten Versuch kennegelernt haben und an dieser Stelle erstmalig einsetzen. Ihre Werkzeuge zur Beurteilung der Leistungsfähigkeit eines neuronalen Netzes werden in diesem Versuch signifikant erweitert: Sie lernen die Kreuzkorrelation, die mittlere quadratische Abweichung, sowie die strukturelle Ähnlichkeit kennen. Nach Bearbeitung dieses Versuchs haben Sie die im Zuge der Digitalisierung immer relevanter werdende Disziplin der Bildrekonstruktion kennengelernt und können diese erfolgreich in neuronalen Netzen umsetzen.

2 Grundlagen

2.1 Einführung zur Multimodefaser

Optische Kommunikationsnetze auf Basis von Glasfasern bilden das Rückgrat der gegenwärtigen Infrastruktur des globalen Datenaustausch. Mit Glasfasern können über spezielle Multiplextechniken wie bspw. dem Wellenlängenmultiplex enorme Datenraten (Übertragung mit Lichtgeschwindigkeit) effizient und auf engstem Raum erreicht werden. Ihre praktischen physischen Eigenschaften (dehn- und biegbar) machen sie zu einem flexibel einsetzbaren Medium zur Datenübertragung [5].

Glasfasern sind rotationssymmetrische, zylindrische Lichtwellenleiter bestehed aus einem Kern, der den Brechungsindex n_1 besitzt und von einem Mantel mit dem Brechungsindex n_2 umgeben ist, wobei $n_1 > n_2$. Charakteristisch für Glasfasern ist ihr Brechnungsindexprofil. Ein weit verbreiteter Glasfasertyp ist die Stufenindexfaser, bei der das Brechnungsindexprofil stufenförmig von den Kern- und Mantelbereich übergeht. Die Lichtleitung erfolgt schließlich nach dem Prinzip der Totalreflexion am Kern-Mantelübergang. In Abbildung 1 ist der Querschnitt einer Glasfaser und ein stufenförmiges Brechungsindexprofil gezeigt. Je nach Kernradius, verwendeter Lichtwellenlänge und Brechungsindexdifferenz zwischen Kern und Mantel (numerische Apertur NA $= \sqrt{n_1^2 - n_2^2}$) können unterschiedlich viele, diskrete Wege des einfallenden Lichtstrahls in zickzack-förmigen Bahnen vorgenommen werden. Sämtliche mögliche Ausbreitungsformen des einfallenden Lichts können als sog. Eigenfunktionen, bzw. *Moden*, der Faser interpretiert werden. Der einfachste, und schnellste Weg ist der direkte Weg über die Hauptachse des Kerns, welcher stellvertretend für den Grundmode ist. Die transversalen Feldverteilungen der einzelnen Moden werden über die Lösung der Wellengleichung nach Maxwell ermittelt. Existieren über geeignete Parameterwahl mehrere mögliche Ausbreitungswege durch die Faser, spricht man von einer *Multimodefaser* [4].

Bei einer hohen Anzahl ausbreitungsfähiger Moden ($N \gg 1$), kann die Multimodefaser auch als bildübertragendes Medium verwendet werden. Aufgrund ihrer hohen Flexibilität werden gegenwärtig Multimodefasern zur Verwendung in der medizinischen Endoskopie erforscht [7]. Ein Lichtsignal, bspw. bei der Beleuchtung einer biologischen Zelle, wird bei der Einkopplung in die Faser in die zur Verfügung stehenden Moden übersetzt. Dabei wandern die einzelnen angeregten Moden durch die Faser. Allerdings vermischen sich die einzelnen Moden auf ihrem Weg und es findet ein Übersprechen (engl.: *crosstalk*) statt. Das Übersprechen ist dabei abhängig von Fertigungstoleranzen oder Biegung der Faser, sowie Unsicherheiten bei der Lichteinkopplung, weshalb das Übersprechen im Allgemeinen als unvorhersehbar gilt. Am Ausgang der Faser interferieren alle angeregten und vermischten Moden zu einem sog. *Specklemuster*. Das Prinzip dieses Phänomens ist in Abbildung 2 dargestellt. Mithilfe moderner computergestützter Ansätze ist es jedoch möglich das Übersprechen zu kompensieren und die Specklemuster zu verarbeiten. Beispielsweise kann

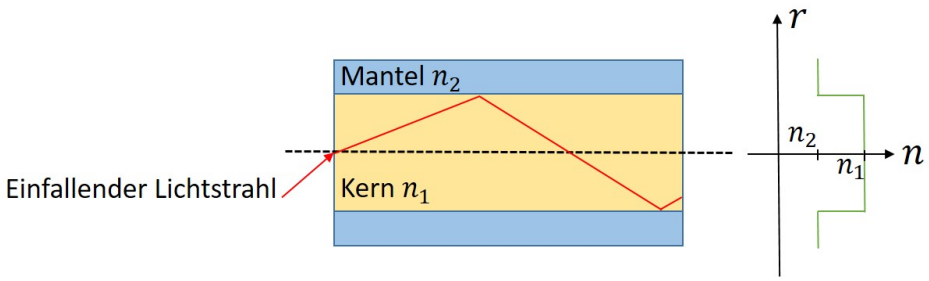

Abbildung 1: Querschnitt einer Glasfaser und stufenförmiges Brechungsindexprofil. Ein einfallender Lichtstrahl bricht hin zur Hauptachse und totalreflektiert am Kern-Mantelübergang. Je nach Parametervariation existiert eine diskrete Anzahl zickzack-förmiger Bahnen, auf denen das Licht seinen Weg durch die Faser nimmt. Jede dieser Bahnen ist stellvertertend für eine sog. Eigenfunktion, bzw. Mode der Glasfaser. Liegt mehr als eine Mode vor, spricht man von einer Multimodefaser.

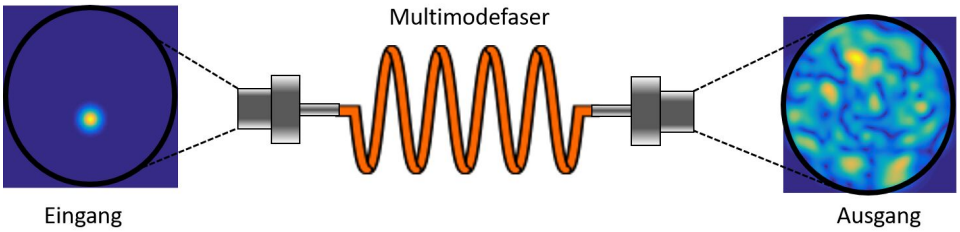

Abbildung 2: Prinzip des Modenübersprechens. Wird eine beliebige Lichtfeldverteilung am Eingang einer Multimodefaser eingekoppelt, wird eine diskrete Anzahl Moden innerhalb der Faser angeregt. Während ihrer Ausbreitung vermischen sich die Moden und es kommt zu einem Übersprechen untereinander. Am Ausgang der Multimodefaser kommt es zur Interferenz und es ist eine granulierte Struktur, ein sogenanntes Specklemuster, sichtbar.

der Zusammenhang zwischen Lichteinkopplung am Fasereingang und Lichtauskopplung am Ausgang von einem neuronalen netz erlernt werden. Ziel ist es, auf Basis eines vorliegenden Specklemusters, die zugehörige Eingangsverteilung zu rekonstruieren (siehe Borhani et al. [3]). Dabei werden Paare von Lichtverteilungen am Eingang mit zugehörigen, vermischten Lichtverteilungen am Ausgang der Multimodefaser als Datensatz für das Training des neuronalen Netzes (NN) verwendet. Dieser Ansatz ist Gegenstand des vorligenden Praktikums.

2.2 Netzarchitekturen für die Bildrekonstruktion

In diesem Abschnitt sollen zwei Architekturen vorgestellt werden, mit denen Regressionsaufgaben gelöst werden können. Im Falle dieses Praktikumsversuchs ist dies die Rekonstruktion von Ziffern aus Specklemustern einer Multi-

modefaser. Zum einen können erneut die aus dem ersten Versuch bekannten MLPs verwendet werden, aber auch die weit verbreitete Netzarchitektur der Convolutional Neural Networks (CNN) in Form des U-Nets wird vorgestellt.

- **Multilayer Perceptron (MLP)**
  Die Netzarchitektur Multilayer Perceptron wurde bereits im ersten Praktikumsversuch zur Klassifizierung von Ziffern verwendet und kann auch bei der Rekonstruktion von Ziffern aus den Specklemustern einer Multimodefaser eingesetzt werden. Während ein Single Layer Perceptron lediglich aus einem Input und einem Output Layer besteht, ist hier zusätzlich (mindestens) ein Hidden Layer vorhanden. Die einzelnen Layer sind fully connected, d.h. die Neuronen eines Layers sind mit allen Neuronen des folgenden Layers verbunden.

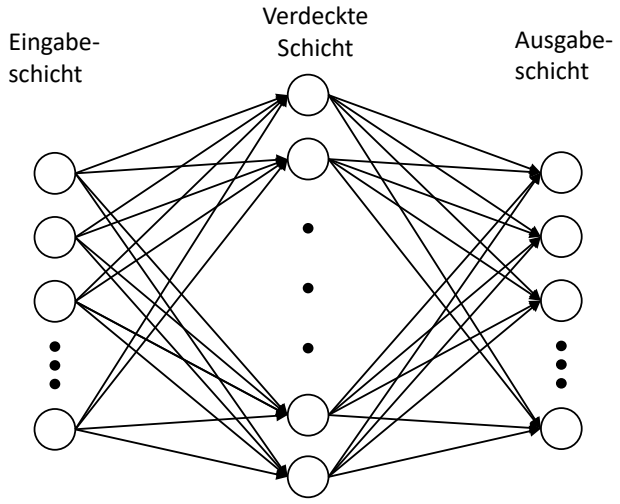

Abbildung 3: Multilayer Perceptron mit einem Hidden Layer

Für die Verwendung eines MLP für eine Regressionsaufgabe mit der Funktion `trainNetwork` wird neben dem bekannten Aufbau aus Image Input Layer, Fully Connected Layern und Aktivierungsfunktionen jetzt am Ausgang zunächst ein `depthToSpace2dLayer` benötigt, um den Ausgang der Fully Connected Layer, welche die Größe $[1,1,h*w*c]$ besitzen, in das gewünschte Format des Ausgangsbildes $[h,w,c]$ zu transformieren. Danach wird ein `regressionLayer` benötigt, welches den gewünschten Output erzeugt. Die Netzarchitektur für Eingangsbilder mit dem Format $[I_{px},I_{px},1]$ und Ausgangsbilder mit dem Format $[O_{px},O_{px},1]$ könnte folgendermaßen aussehen:

```
layers = [imageInputLayer([I_px I_px 1],...
   'Name','Input')
   fullyConnectedLayer(I_px^2,'Name','Fc1')
   reluLayer('Name','Relu1')
```

```
    fullyConnectedLayer(O_px^2,'Name','Fc2')
    reluLayer('Name','Relu2')
    depthToSpace2dLayer([O_px O_px],'Name','dts1')
    regressionLayer('Name','Output')
];
```

Für die Verwendung einer eigenen Trainingsschleife ist das Vorgehen etwas komplizierter, als im ersten Versuch. Da bei der Verwendung für ein `dlnetwork` der Ausgang eines Fully Connected Layers nur eindimensional ist, müssen die Dimensionen zur Verwendung des `depthToSpace2dLayer` vorher wieder hinzugefügt werden. Dafür ist eine eigene Matlab-Klasse für das Layer `addDimensionsLayer` beigefügt, welches diese Aufgabe erfüllt. Wie schon bei den Netzen zur Klassifizierung kann auf das Output Layer (in diesem Fall `regressionLayer`) verzichtet werden, wenn eine eigene Trainingsschleife verwendet wird. Eine beispielhafte Netzarchitektur könnte so aussehen:

```
layers = [
    imageInputLayer([I_px I_px 1],'Name','Input')
    fullyConnectedLayer(I_px^2,'Name','Fc1')
    reluLayer('Name','Relu1')
    fullyConnectedLayer(O_px^2,'Name','Fc2')
    reluLayer('Name','Relu2')
    addDimensionsLayer('SS','Name','adl1')
    depthToSpace2dLayer([O_px O_px],'Name','dts1')
];
```

- **U-Net**
  Das U-Net gehört zu den Convolutional Neural Networks und wurde ursprünglich für die Bildsegmentierung in der Biomedizin entwickelt [6]. Die Bildsegmentierung ist der Prozess der Aufteilung eines Bildes in einzelne Segmente, die jeweils aus einer Menge an Pixeln bestehen und Klassen zugeordnet werden. So können z.B. Objekte oder Grenzen in Bildern lokalisiert werden, indem jedem Pixel ein bestimmtes Label zugeordnet wird.

  Das U-Net besteht aus einem kontrahierenden (Encoder) und einem expansiven (Decoder) Pfad. Der kontrahierende Pfad folgt dabei der typischen Architektur von Convolutional Neural Networks und besteht aus mehreren Stufen von zwei 3x3 Convolutional Layern (unpadded), die jeweils von einem ReLU Layer gefolgt werden. Danach wird ein 2x2 Max Pooling Layer mit einem Stride von 2 für das Downsampling platziert. Beim Downsampling wird jeweils die Auflösung der Bilder verkleinert,

aber die Anzahl der Feature Channels verdoppelt. In jeder Stufe des expansiven Pfades findet dann ein Upsampling der Feature Channels statt, welche von einer 2x2 Convolution (Up-Convolution) gefolgt wird, welche die Anzahl der Feature Channels wieder halbiert und die Auflösung vergrößert. Darauf folgt eine Verbindung dieser Feature Channels mit der zugehörigen Feature Channels des Encoders mithilfe einer Skip Connection. Danach werden zwei 3x3 Convolutional Layer mit jeweils einem ReLU Layer platziert [6].

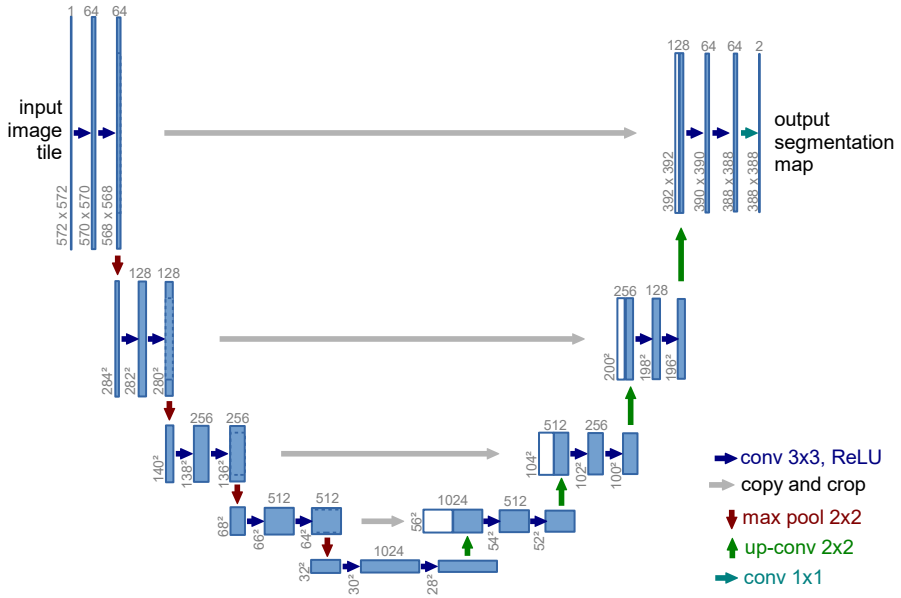

Abbildung 4: U-Net Architektur [6]

Für die Verwendung des U-Nets in MATLAB existiert die eigene Funktion `unetLayers`, welche ein vollständiges U-Net für die Bildsegmentierung erstellt. Dieses kann anschließend so modifiziert werden, dass es für eine beliebige Regressionsaufgabe geeignet ist. Eine funktionierende Netzarchitektur für das Training mit `trainNetwork` kann z.B. folgendermaßen erreicht werden (hier muss die Auflösung des Ausgangsbildes der Auflösung des Eingangsbildes entsprechen):

```
layers = unetLayers([I_px I_px 1],2,...
    'encoderDepth',3);
finalConvLayer = convolution2dLayer(1,1,...
   'Padding','same','Stride',1,'Name',...
   'Final-ConvolutionLayer');
layers = replaceLayer(layers,...
   'Final-ConvolutionalLayer',finalConvLayer);
layers = removeLayers(layers,'Softmax-Layer');
regLayer = regressionLayer('Name','Reg-Layer');
layers = replaceLayer(layers,...
```

```
    'Segmentation-Layer',regLayer);
layers = connectLayers(layers,...
    'Final-ConvolutionLayer','Reg-Layer');
```

Für benutzerdefinierte Training-Schleifen muss die Architektur entsprechend angepasst werden:

```
layers = unetLayers([I_px I_px 1],2,...
    'encoderDepth',3);
finalConvLayer = convolution2dLayer(1,1,...
    'Padding','same','Stride',1,'Name',...
    'Final-ConvolutionLayer');
layers = replaceLayer(layers,...
    'Final-ConvolutionalLayer',finalConvLayer);
layers = removeLayers(layers,'Softmax-Layer',...
    'Segmentation-Layer');
```

2.3 Overfitting

Unter Overfitting versteht man die Überanpassung eines Netzes an die vorgegebenen Trainingsdaten. Dadurch ist das Netz nicht mehr in der Lage, zu generalisieren. Das kann zum einen an einem Mangel an unterschiedlichen Trainingsdaten liegen. Umgekehrt ist aber auch möglich, dass zu viele lernbare Parameter existieren. Während des Trainingsprozesses wird Overfitting sichtbar, wenn die Verlustfunktion (im Falle der Regression der MSE) für die Trainingsdaten kleiner wird, aber der Verlust für die Validierungsdaten nicht mehr abnimmt. In diesem Fall divergieren beide Kurven voneinander. Um diesem Effekt entgegenzuwirken, können verschiedene Maßnahmen ergriffen werden, die im folgenden Abschnitt beschrieben werden.

- **Early Stopping**
  Overfitting kann durch ein zu langes Training entstehen. Dabei lernt das Netz zunächst Validierungsdaten korrekt zu verarbeiten, erreicht aber irgendwann eine Sättigungsphase, nach der die Rekonstruktionsgenauigkeit der Validierungsdaten wieder abnimmt, während nur die Genauigkeit für die Trainingsdaten erhöht wird. Das Modell merkt sich also nur noch Beispiele, anstatt zu lernen, bestimmte Merkmale (Features) zu erkennen. In MATLAB ist es für das Training mit `trainNetwork` mit dem Parameter `ValidationPatience` möglich, eine Anzahl zu spezifizieren, wie oft die Verlustfunktion der Validierungsdaten größer oder gleich des bisher kleinsten Verlustes sein kann, bevor das Training abgebrochen wird.

  ```
  options.ValidationPatience = 100;
  ```

- **Data Augmentation**
  Um ein neuronales Netz zu trainieren, welches in der Lage ist, gut zu generalisieren, ist eine geeignete Auswahl des Trainingsdatensatzes essenziell. Diese Daten sollten daher möglichst umfangreich sein und alle möglichen Fälle abbilden. Da in der Realität meist nur sehr begrenzte Trainingsdaten vorliegen, können verschiedene Techniken zur Data Augmentation angewendet werden. Im Wesentlichen bedeutet das, die Menge der Trainingsdaten zu erhöhen, indem modifizierte Kopien existierender Daten bzw. synthetische Daten aus exisitieren Trainingsdaten verwendet werden. Für Bilder können Operationen wie Spiegelung, Rotation, Verschiebung oder andere geometrische Transformationen verwendet werden, um mehr Trainingsdaten zu erzeugen. In MATLAB existieren Befehle wie `imrotate` zur Rotation oder `circshift` zur Verschiebung, die solche Operationen ermöglichen. Dabei muss darauf geachtet werden, dass die Auflösung der Bilder nicht verändert wird.

- **Dropout Layer**
  Ein Dropout Layer sorgt dafür, dass einzelne Eingangswerte eines Layers während des Trainings mit einer angegebenen Wahrscheinlichkeit auf Null gesetzt werden. Für tiefe neuronale Netze mit vielen Parametern sind diese Layer ein wichtiges Mittel, um Overfitting zu reduzieren. In MATLAB kann ein Dropout Layer folgendermaßen erzeugt werden:

  ```
  probability = 0.5;
  layer = dropoutLayer(probability);
  ```

  Im Falle der oben vorgestellten Architektur U-Net sind von MATLAB bereits zwei Dropout Layer integriert worden.

2.4 Evaluierung von Regressionsnetzen

Nachdem ein neuronales Netz trainiert worden ist, muss es getestet werden, um seine Generalisierungsfähigkeit zu evaluieren. Ein erfolgreich trainiertes neuronales Netz erbringt gute Ergebnisse bei ungesehenen Daten. Bei Klassifizierungsproblemen ist die Genauigkeit der Vorhersageergebnisse ein intuitives Bewertungskriterium. Für die Bewertung von Regressionsmodellen, bei denen der Output ein kontinuierlich variierender Wert ist, gibt es häufig unterschiedliche Kriterien. Bei diesem Versuch ist der wahre Wert (ground truth) der Testdaten verfügbar. Daher kann die Techniken *full-reference image quality assessment* (FR-IQA) verwendet werden. Die folgenden Methoden werden üblicherweise verwendet.

- **RMSE**
  Die Wurzel der mittleren Fehlerquadratsumme (root-mean-square error, RMSE) ist ein häufig verwendetes Maß für die Unterschiede zwischen

den von einem Modell oder einem Schätzer vorhergesagten Werten und den wahre Werten. Der RMSE kann wie folgt berechnet werden:

$$RMSE = \sqrt{\frac{1}{h \times w} \Sigma_i^{h \times w} \Big(R_i - P_i\Big)^2}. \tag{1}$$

h und w sind die Höhe und Breite des Bildes. R ist das reale Referenzbild, während P das vorhergesagte Bild ist. Es ist erwähnenswert, dass der Bereich des RMSE in $[0, \infty)$ liegt. 0 bedeutet, dass die beiden Bilder gleich sind.

- **Korrelationskoeffizient**
  Der Korrelationskoeffizient, auch Produkt-Moment-Korrelation, ist ein Maß für den Grad des linearen Zusammenhangs zwischen zwei mindestens intervallskalierten Merkmalen, das nicht von den Maßeinheiten der Messung abhängt und somit dimensionslos ist [10]. Die Berechnung einer Korrelation lautet wie folgt:

$$CC = \frac{\sum_i^{h \times w} (R_i - \overline{R})(P_i - \overline{P})}{\sqrt{\sum_i^{h \times w} (R_i - \overline{R})^2 \sum_i^{h \times w} (P_i - \overline{P})^2}}, \tag{2}$$

  wobei $\overline{R}$ und $\overline{P}$ die jeweiligen Mittelwerte sind. Der Korrelationskoeffizient kann Werte zwischen -1 und $+1$ annehmen. $+1$ bedeutet, dass die beiden Bilder vollkommen positiv korreliert sind. Es ist wichtig zu beachten, dass der Korrelationskoeffizient unabhängig vom Maßstab der beiden Bilder ist und sich nur auf die Beziehung zwischen den Merkmalen bezieht.

- **SSIM**
  Die strukturelle Ähnlichkeit (englisch structural similarity, SSIM) wird zur Messung der Ähnlichkeit zwischen zwei Bildern verwendet [8]. Es handelt sich dabei um einen intuitiven Standard für die Bewertung der Qualität eines Bildes, der darauf abzielt, es mit der menschlichen Wahrnehmung in Einklang zu bringen.

$$SSIM = \frac{(2\mu_R \mu_P + C_1)(2\sigma_{R,P} + C_2)}{(\mu_R{}^2 + \mu_P{}^2 + C_1)(\sigma_R{}^2 + \sigma_P{}^2 + C_2)}, \tag{3}$$

  mit

  - μ: dem Mittelwert
  - σ^2: der Standardabweichung
  - $\sigma_{R,P}{}^2$: der Kovarianz von R und P
  - $C_1 = (k_1 L)^2$ und $C_2 = (k_2 L)^2$: zwei Variablen zur Stabilisierung der Division bei kleinen Nennern (wenn $\mu_R{}^2 + \mu_P{}^2$ oder $\sigma_R{}^2 + \sigma_P{}^2$ sehr nahe bei 0 liegen). L ist der Dynamikbereich der Pixelwerte (255 für 8-Bit-Graustufenbilder), und k sind zwei kleine Konstante ($k_1 = 0.01, k_2 = 0.03$).

Der SSIM-Index ist ein dezimaler Wert zwischen 0 und 1. Der Wert 1 bedeutet, dass die zwei Bilder identisch sind.

- **PSNR**
  Der Spitzen-Signal-Rauschabstand (peak signal-to-noise ratio, PSNR) ist ein mathematisches Bewertungskriterium für die Bildqualität. Ein höherer PSNR-Wert bedeutet, dass das verzerrte Bild näher am Referenzbild liegt, d. h. die Bildqualität ist besser. PSNR (in dB) kann mit Hilfe von MSE berechnet werden:

$$PSNR = 20 \cdot \log_{10}(\frac{\text{MAX}}{\sqrt{\text{MSE}}}), \tag{4}$$

  mit MAX der maximale Signalwert, der in Referenzbild vorhanden ist. Wenn die Pixel mit 8 Bits pro Abtastung dargestellt werden, ist MAX 255.

- **Box-Plot**
  Nach der Auswahl der geeigneten Evaluierungskriterien ist es wichtig die Ergebnisse zu visualisieren. Der Box-Plot ist ein Diagramm, das zur grafischen Darstellung einer Reihe von Datenverteilungsinformationen verwendet wird. Ein Beispiel ist ein zufälliger Datensatz. Angenommen, es existieren drei Sätze von Testdaten, die jeweils zehn Testbilder enthalten. Die Ergebnisse der Visualisierung sind in der Abb. 5 dargestellt.

```
template = rand(10,3);
figure;
boxchart(template); ylabel('SSIM');
```

  Die Verteilung der Ergebnisse kann anhand des Box-Plots veranschaulicht werden. Die Box entspricht dem Bereich, in dem die mittleren 50% der Daten liegen. Sie wird also durch das obere und das untere Quartil begrenzt. Die Länge der Box entspricht dem Interquartilsabstand. Die Definition des Bereiches ohne Ausreißer (benannte als Whisker) ist nicht einheitlich. Eine mögliche häufige verwendete Definition besteht darin, die Länge der Whisker auf maximal das 1.5-Fache des Interquartilsabstands (1.5×IQR) zu beschränken. Häufig werden Werte, die außerhalb von 1.5×IQR liegen, als Ausreißer bezeichnet [9]. Ein kleiner Abstand zwischen Box und Whisker ohne Ausreißer bedeutet bessere Testergebnisse, die auch auf ein robustes neuronales Netzmodell hindeuten.

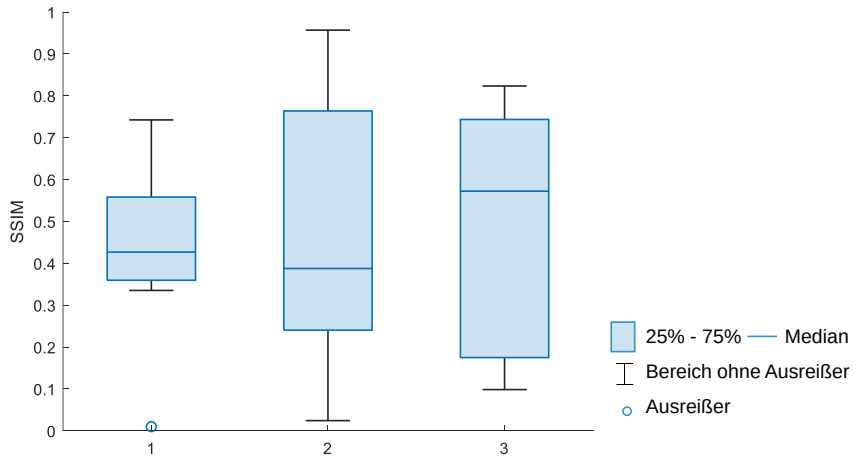

Abbildung 5: Box-Plot Beispiel

3 Fragen zur Versuchsvorbereitung

Bevor Sie mit der Bearbeitung der Aufgaben starten, beantworten Sie als Vorbereitung bitte folgende Fragen:

1. Was ist der Unterschied zwischen Klassifizierung und Regression?
2. Woran kann erkannt werden, dass Overfitting beim Training eines NN vorliegt?
3. Mit welchen Maßnahmen kann Overfitting vorgebeugt, bzw. mit welchen Maßnahmen kann Overfitting entgegengewirkt werden?
4. Wozu dienen die „Skip-Connections“ in einem U-Net?
5. Mithilfe einiger der aufgeführten Gütekriterien RMSE, Kreuzkorrelation und SSIM, kann die „Ähnlichkeit“ zweier Bilder beurteilt werden und mit manchen nicht. Mit welchen dieser Gütekriterien ist dies möglich und weshalb? Begründen Sie anhand der aufgeführten Formeln.
6. Erläutern Sie die einzelnen Intervalle, die in einem Box-Plot eingezeichnet werden und inwiefern mit einem Boxchart die Robustheit eines NN beurteilt werden kann.

4 Aufgaben zur Versuchsdurchführung

Im Rahmen des Praktikums sollen folgende Aufgaben bearbeitet werden:

1. Benutzen Sie das MATLAB Skript `train_reconstruction_net.m` und binden Sie den vorbereiteten Datensatz `DATA_MMF_16.mat` für die Bildrekonstruktion einer Multimodefaser ein. Nutzen Sie hierfür geeignete Befehle zum Laden eines Datensatzes. Sie erhalten Datensätze für Trainings-, Validierungs- und Testdaten.

2. Erstellen Sie ein MLP zur Bildrekonstruktion. Beachten Sie, dass das Netz für Regressionsaufgaben geeignet sein muss. Achten Sie auf die benötigte Auflösung der Eingangs- und Ausgangsbilder.

3. Definieren Sie passende Hyperparameter für den Trainingsvorgang. Setzen Sie den Parameter `validationPatience` auf einen geeigneten Wert. Hiermit stoppt Ihr Trainingsvorgang automatisch, sobald Overfitting auftritt und sich der Validierungswert nach der jeweiligen Anzahl an Iterationen nicht verbessert hat. Starten Sie anschließend den Trainingsvorgang.

4. Sie werden feststellen, dass extremes Overfitting auftritt. Ihre Aufgabe ist es nun, den bereitgestellten Trainingsdatensatz entscheidend zu erweitern, um dem Overfitting entgegenzuwirken. Hierzu müssen Sie die bereitgestellten Eingangsbilder (Ziffern) bearbeiten. Nutzen Sie das Skript `create_augmented_data_mmf.m`, um den Datensatz zu verdreifachen, indem Sie aus jedem einzelnen Eingangsbild zwei neue Bilder erzeugen. Verwenden Sie hierzu Methoden der Data Augmentation (siehe Abschnitt 2.3). Erzeugen Sie auf Basis der bearbeiteten Eingangsbilder, zugehörige Ausgangsbilder, indem Sie die Funktion `mmf()` verwenden. Die Funktion wird Ihnen passende Eingangs- und Ausgangsbilder liefern.

5. Wiederholen Sie den Trainingsvorgang unter Verwendung des erweiterten Datensatzes. Auch hier wird es nicht möglich sein, Overfitting zu verhindern, aber der Unterschied zwischen Trainings- und Validationsdaten sollte deutlich kleiner werden. Der mittlere Kreuzkorrelationskoeffizient sollte sich durch Data Augmentation in etwa von $40\% - 50\%$ auf $70\% - 80\%$ verbessern.

6. Evaluieren Sie die trainierten Netze mithilfe der Testdaten und vergleichen Sie die Ergebnisse in geeigneten Boxplots mithilfe der Gütekriterien SSIM, Kreuzkorrelation, RMSE und PSNR.

7. Erstellen Sie ein passendes U-Net zur Bildrekonstruktion. Verwenden Sie zum Training den erweiterten Trainingsdatensatz.

8. Stellen Sie die Ergebnisse des MLP und des U-Net (beide trainiert mit dem erweiterten Datensatz) in einem geeigneten Box-Plot dar. Vergleichen Sie die Performance beider Netze mit Hinblick auf Genauigkeit, Unsicherheit (Standardabweichung) und benötigte Trainingszeit. Nach wie vielen Epochen konvergieren die beiden Netze? Setzen Sie hierfür ein definiertes Limit mit dem Hyperparameter `validationPatience`.

Literatur

[1] Charu C Aggarwal u. a. “Neural networks and deep learning”. In: *Springer* 10 (2018), S. 978–3.

[2] Visible Body. *Gehirn und Nerven: 5 Schlüssel öffnen das Nervensystem*. https://www.visiblebody.com/de/learn/nervous/system-overview. [Online; accessed 11-November-2021]. 2021.

[3] Navid Borhani u. a. "Learning to see through multimode fibers". In: *Optica* 5.8 (2018), S. 960–966.

[4] Detlef Gloge. "Weakly guiding fibers". In: *Applied optics* 10.10 (1971), S. 2252–2258.

[5] Benjamin J Puttnam, Georg Rademacher und Ruben S Luus. "Space-division multiplexing for optical fiber communications". In: *Optica* 8.9 (2021), S. 1186–1203.

[6] Olaf Ronneberger, Philipp Fischer und Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: (2015), S. 234–241.

[7] Sergey Turtaev u. a. "High-fidelity multimode fibre-based endoscopy for deep brain in vivo imaging". In: *Light: Science & Applications* 7.1 (2018), S. 1–8.

[8] Zhou Wang u. a. "Image quality assessment: from error visibility to structural similarity". In: *IEEE transactions on image processing* 13.4 (2004), S. 600–612.

[9] wikipedia. *Box-Plot*. https://de.wikipedia.org/wiki/Box-Plot. [Online; accessed 23-November-2021]. 2021.

[10] wikipedia. *Korrelationskoeffizient*. https://de.wikipedia.org/wiki/Korrelationskoeffizient. [Online; accessed 23-November-2021]. 2021.