Zeitreihenvorhersagen mit sequentiellen neuronalen Netzwerken

Bachelorseminar (Inf, WInf)

Prof. Dr. Sören Pirk
Visual Computing and Artificial Intelligence Group
sp@informatik.uni-kiel.de

Motivation

Dieses Seminar adressiert das Lernen von Modellen für die Analyse und Vorhersage von Zeitreihen. Zeitreihen sind temporär geordnete Datenpunkte, die z.B. stündlich, wöchentlich oder jährlich erhoben werden. Die Analyse und das Vorhersagen von Zeitreihen sind für eine Vielzahl verschiedener Bereiche (z.B. Wirtschaft, Fernerkundung, Robotik, Meteorologie) von großem Interesse.

Mit Modellen für Zeitreihen soll das Vorhersagen von Trends ermöglicht werden, die es erlauben, Aussagen zur zukünftigen Entwicklung von Beobachtungen abzuleiten. Ziel des Seminares ist es, Modelle für zeitlich organisierte Daten mit Hilfe von sequentiellen neuronalen Netzwerken zu lernen. Im Seminar werden verschiedene Architekturen von neuronalen Netzwerken vorgestellt (LSTM, ConvNets, Transformer) und deren Möglichkeiten zur Analyse und Vorhersage von Zeitreihen untersucht.

Die Veranstaltung zielt insbesondere darauf ab, Kompetenzen im Umgang mit realen Daten und der Implementierung von sequentiellen neuronalen Netzwerkarchitekturen zu erwerben. Gerade in Bezug auf praktische Anwendungsfälle sollen Kompetenzen erworben werden, die es den Studierenden ermöglichen, eigene Modelle für die Zeitreihenvorhersage zu trainieren und zu verwenden. Im Rahmen des Seminars bearbeiten die Studierenden, alleine oder in Zweiergruppen, eine zeitreihenanalytische Fragestellung mit dem Ziel, ein eigenes, sequentielles Modell zu entwickeln und zu trainieren.

Organisatorische Angaben

Seminar, 2 SWS, benoteter Schein, ECTS-Studium, ECTS-Credits: 5, Präsenzveranstaltung

Aufgabenstellung und Prüfungsleistung

Die Resultate der analysierten Daten und der trainierten Modelle werden schriftlich ausgearbeitet und am Ende der Vorlesungszeit dem Kurs präsentiert. Die Ausarbeitung und Präsentation werden als Prüfungsleistung bewertet.

Hinweis: Beim Trainieren von neuronalen Netzwerken für echte Anwendungen geht es oftmals um das Finden von kreativen Lösungen bei der Entwicklung von Netzwerkarchitekturen oder dem Organisieren von Daten. Eine Absicht des Seminars ist es daher, möglichst realistisch an das Trainieren von neuronalen Netzwerken heranzugehen. Die Problemstellung ist daher teilweise unterspezifiziert. Es geht weniger um “Richtig oder Falsch” im Umgang mit der Aufgabenstellung, sondern eher um die Vorgehensweise bei der Lösungsfindung.

Technische Vorbereitung

Das Ziel des Seminars ist es, ein neuronales Netzwerk für das Vorhersagen von Zeitreihen zu trainieren. Code für das Implementieren und Trainieren von neuronalen Netzwerken soll entweder mit Google Colab oder mit einer lokalen Python Installation erzeugt werden. Für Colab benötigst du einen Google Account. Um einige der vorgeschlagenen Datensätze herunterzuladen, brauchst du zusätzlich noch einen Kaggle-Account (link).

Um lokal auf deinem Rechner zu arbeiten (und nicht im Browser) musst du deine eigene Python Umgebung mit den entsprechenden APIs (e.g. Keras/Pytorch) für das Trainieren von neuronalen Netzwerken einrichten. Ein Tutorial dafür findest du z.B. hier: Pytorch, Keras.

Um direkt die bereitgestellten Code-Bespiele verwenden zu können, sollte mit Mini Conda und Tensorflow/Keras gearbeitet werden. Hierzu sollte zunächst Miniconda (link) installiert werden. Nach der Installation kann die Umgebung mit den folgenden Befehlen in einer command shell aufgesetzt werden:

```
conda create --name <name> python==3.8
conda activate <name>

conda install -c conda-forge cudatoolkit=11.2 cudnn=8.1.0

pip install tensorflow-gpu==2.5
pip install matplotlib==3.6.2
```

Beispiel-Code

Beispiel-Code für das Implementieren von sequentiellen neuronalen Netzwerken und eines Frameworks für das Trainieren findest du hier: CoLab(link), Files (link).

Datensätze

Es wird empfohlen, einen der folgenden Datensätze für die Erarbeitung der Aufgaben zu verwenden. Gerne kannst du auch einen selbst erzeugten Datensatz oder einen eigens gefundenen Datensatz verwenden. Sollte das der Fall sein, sprich das vorher bitte kurz per E-Mail ab, um die Ein- und Ausgaben und Aufgabenstellungen zu definieren.

- **Climate Change: Earth Surface Temperature Data (link)**
- **Thör Dataset - Tracking Human Motion at Örebro University (link)**
- **Store Item Demand Forecasting Challenge (link)**
- **Appliances Energy Prediction (link)**
- **Weather Prediction (link)**
- **Wildfire Prediction (link)**

Netzwerkarchitekturen

Im Rahmen des Seminars sollen verschiedene Netzwerkarchitekturen untersucht werden. Speziell sollen die folgenden Architekturen zum Einsatz kommen:

- MLP
- 1D Conv Net
- LSTM
- Transformer

Aufgaben und Fragestellungen

Phase 1

In Phase 1 geht es um das:

- Einrichten von Google Colab und/oder nativem Python Setup.
- Erzeugen eines Datensatzes: Sinus-Funktion als Zeitreihen (Beispiel-Code) von 500 Datenpunkten.
- Bekanntmachen mit dem Framework (Beispiel-Code).
- Trainieren auf Sinus-Test-Datensatz.
- Explorieren verschiedener Netzwerk-Architekturen (MLP, Conv1D, LSTM, Transformer).

Folgende Fragestellungen sind zu bearbeiten:

- Welche Architektur performt am besten (qualitativ, quantitativ)?
- Wie lange dauert es, mit den verschiedenen Architekturen eine Zeitreihe von 1000 Steps vorherzusagen?
- Was passiert bei der Vorhersage von Zeitreihen, wenn das Netzwerk mit und ohne 'Data Augmentation' trainiert wird?
- Wie verhält sich die Performance einer beliebigen Architektur, wenn die Länge der Input-Sequenzen von 8, 16, 32, 64, verändert wird?
- Was kann über die Loss-Kurven einer beliebigen Architektur ausgesagt werden, wenn die Anzahl der Weights halbiert/verdoppelt wird?
- Implementiere eine Funktion, die den durchschnittlichen Fehler (Ground Truth vs. Vorhersage) für eine Zeitreihe von 1000 Steps misst.

Phase 2

In Phase 2 geht es um:

- Bekannt machen mit einem oder mehrerer der 6 Datensätze.
- Analyse der Daten und Organisieren der Inputs/Outputs.
- Trainieren mit allen NN-Architekturen.
- Implementierung verschiedener Visualisierungen für die Analyse der Daten/Predictions.
- Training und Analyse auf den Daten.

Folgende Fragestellungen sind zu bearbeiten:

- Implementiere eine Data-Loader Pipeline für den von dir ausgewählten Datensatz. Die Daten-Pipeline soll Sequenzen von Datenpunkten als Mini-Batches (wie im Code-Beispiel gezeigt) bereitstellen.

- Implementiere Code für das Visualisieren der Daten (e.g., Histogramme, t-SNE Plots). Welche Daten der Datensätze können wie untersucht und visualisiert werden?
- Welche Architektur performt am besten (qualitativ, quantitativ)?
- Normalisiere die Daten vor dem Trainieren deines Netzwerks und vergleiche die Vorhersagegenauigkeit, wenn das Netzwerk mit oder ohne normalisierte Daten trainiert wird.
- Wie verhält sich die Performance einer beliebigen Architektur, wenn die Länge der Input-Sequenzen von 8, 16, 32, 64, verändert wird?

Phase 3

In Phase 3 geht es um:

- Dokumentation von Herangehensweise, Aufgaben und Experimenten.
- Wissenschaftliches Vorgehen und Schreiben.

Aufgaben:

- Schreibe ein 4-6 Seiten Paper: Double Column mit Introduction, Method, Datensatz, Training-Procedure, Results, Conclusion (Chat-GPT darf helfen). Es soll das ACM Conference Proceedings Template verwendet werden: link.
- Präsentiere deine Ergebnisse und beantworte Fragen zu dem von dir ausgewählten Datensatz und deiner Vorgehensweise.

Literatur

- Deep Learning, Ian Goodfellow and Yoshua Bengio and Aaron Courville, 2016, MIT Press

Fragen/Probleme

Bei Fragen oder Problemen schreib eine E-Mail an sp@informatik.uni-kiel.de.