Zitierguide Programmieren

Umgang mit (Python-)Programmcode in Hausarbeiten

1. Python – Module/Programmbibliotheken

Man sollte immer im Fließtext angeben, mit welcher Python-Version man gearbeitet und welche Module bzw. Programmbibliotheken man verwendet hat. Für Module, die nicht zur Python Standardbibliothek gehören, bedarf es außerdem einer eigenen Quellenangabe. (Die Liste der Standard-Module findet man auf der Python Dokumentationsseite.) Bspw. sind *spaCy*, *stanza*, *pandas*, *numpy*, *requests*, *scrapy*, *seaborn* oder *sklearn* von Python separate Bibliotheken, die Module *math*, *random*, *os*, oder *re* in Python hingegen nicht.
Falls es eine Referenzpublikation für die Programmbibliothek gibt, sollte diese vorrangig zitiert werden. Für die Bibliothek *pandas*, die häufig für die Verwaltung und Analyse von (vor allem tabellarischen) Daten verwendet wird, würde man z. B. folgende Quelle angeben:

> McKinney, Wes (2010). „Data Structures for Statistical Computing in Python." In: Proceedings of the 9th Python in Science Conference. Austin, Texas, S. 56–61, DOI: 10.25080/Majora-92bf1922-00a.

Die entsprechende Publikation findet man i.d.R., wenn man im Internet nach „how to cite [Name der Programmbibliothek]" sucht, eine Hilfestellung bietet hier aber auch die Seite Cite Bay. Wenn eine Referenzpublikation nicht ermittelt werden kann, sollte eine einschlägige URL angegeben werden, unter der das Paket verfügbar gemacht und/oder dokumentiert wird, z. B. die offizielle Dokumentationsseite.

2. Programmcode

a. Eigener Code

Kleinere Ausschnitte des für die Arbeit entwickelten Programmiercodes oder ggf. eigens aufbereitete oder erschlossene Daten (bspw. in XML oder JSON) können in der Arbeit zitiert und kommentiert werden. Dies bietet sich insbesondere für solche Code-Passagen an, die für die Arbeit entscheidend sind, bei denen besondere Schwierigkeiten überwunden oder eine besonders elegante Lösung entwickelt wurde. Es gibt von der Professur keine festen Vorgaben dafür, wie man auf den eigenen Code in einer Hausarbeit verweisen soll, die folgenden Vorschläge sollen daher als Orientierung dienen.
Eine Möglichkeit ist, die beschriebene Stelle mit Zeilenabgaben zu referenzieren.
Beispiel:

> „Für jedes Wort wird zusätzlich hinterlegt, in welchen Dateien es gefunden wurde (54–63[1])."

Ob man die Zeilen wie in dem obigen Beispiel in Klammern angibt oder in Fußnoten sollte an den allgemeinen Zitierstil, den man in der Arbeit verfolgt, angepasst werden.

---

[1] Die Verweise in den Klammern in diesem Abschnitt beziehen sich auf die Zeilen des Skriptes [Angabe wo das Skript zu finden ist, Details s. Abschnitt „Abgabe des Skripts"].

Bei kürzeren Programmskripten reicht es auch, einzelne Funktionen zu referenzieren, indem man auf den Funktionsnamen verweist.
Eine weitere Möglichkeit, die sich vor ebenfalls vor allem bei kompakteren Skripten eignet, ist, den eigenen Code direkt einzufügen, entweder als Abbildung, beispielsweise als Screenshot oder direkt im Dokument als Blockzitat. Längere Code-Beispiele oder ausführlichere Auszüge aus einem Datensatz sollten in den Anhang der Arbeit verlagert werden. Bei Beidem sollte Folgendes beachtet werden:

- Monospace-Schriftart nutzen (z. B. Courier New, Consolas)
- Abbildung/Textfeld nummerieren und beschriften
- ggf. Zeilennummern (analog zum Skript) einfügen

Beispiel Screenshot:

```
5   #Funktion zum Auslesen & extrahieren der .txt-Datei
6   def importNetwork(filename):
7       global articleLinks, articleSizes, nodeList
8
9       fp = open(filename, "r", encoding="utf-8")
10      line = fp.readline()
11      while line:
12          key = line.split("=>")[0].split("||")[0] #Artikelname extrahieren
13          size = line.split("=>")[0].split("||")[1] #Zahl (Artikelgröße) extrahieren
14          articleLinks[key] = []        #Liste in Dic initialisieren, key = aktueller Knoten
15          articleSizes[key] = size      #Größe einfügen
```

Abbildung 1: Funktion zum Auslesen und Extrahieren der .txt-Datei (Z. 5-15)

Um Code mit Syntax-Highlighting direkt ins Dokument einzufügen, gibt es zwei Möglichkeiten. Mit *LibreOffice Writer* und *MS Word* kann unter „Einfügen“ ein Textfeld mit der Maus erstellt werden, in das der Code einfach hereinkopiert wird. Das Syntax-Highlighting wird dabei direkt übernommen, wenn man entweder die IDE Visual Studio Code nutzt (Skript öffnen, markieren, kopieren), oder den Editor Notepad++ (Skript öffnen, Plugins > NppExport > Copy all formats to clipboard).

In *MS Word* gibt es noch die zweite Option, ein Dokument im Dokument zu erstellen. Dazu auf Einfügen > Objekt > Neu erstellen > Objekttyp: MS Word Document > OK. Nun geht entweder automatisch oder durch Anklicken des Bereichs ein weiteres Word-Dokument auf, in das der Code hereinkopiert werden kann. Die Größe passt sich automatisch an den Inhalt an, das Syntax-Highlighting wird wie beim Textfeld übernommen.

Sowohl im Textfeld als auch im Word-Objekt können nun weitere Feineinstellungen wie die Anpassung der Schriftart gemacht werden. Um Zeilennummern hinzuzufügen, ist die einfachste Möglichkeit, die Nummerierungsfunktion zu verwenden. Unter den Optionen kann hier angepasst werden, welches Zahlenformat und welcher Schrifttyp verwendet werden soll, und der Nummerierungswert kann festgelegt werden.

Beispiel Textfeld:

```
5   #Funktion zum Auslesen & extrahieren der .txt-Datei
6   def importNetwork(filename):
7       global articleLinks, articleSizes, nodeList
8
9       fp = open(filename, "r", encoding="utf-8")
10      line = fp.readline()
11      while line:
12          key = line.split("=>")[0].split("||")[0] #Artikelname extrahieren
13          size = line.split("=>")[0].split("||")[1] #Zahl (Artikelgröße) extrahieren
14          articleLinks[key] = []        #Liste in Dic initialisieren, key = aktueller Knoten
15          articleSizes[key] = size      #Größe einfügen
```

Abbildung 2: Funktion zum Auslesen und Extrahieren der .txt-Datei (Z. 5-15)

b. Umgang mit Fremdcode

Die Regeln für das Zitieren von Programmcode sind weniger eindeutig und streng als die für Literatur, allerdings gilt auch hier, dass man fremdes Gedankengut stets als solches kennzeichnen sollte.
Ideen und Programme, die – zumindest unter Programmierern – als „Allgemeinwissen“ betrachtet werden, wie z. B. eine einfach for- oder while-Schleife, die bis 100 zählt, müssen i.d.R. nicht referenziert oder zitiert werden. Gibt es nur einen Weg, eine bestimmte Aufgabe zu programmieren (und dieser Weg ist weit verbreitet), muss er auch nicht zitiert werden. Hat man nur die für die Aufgabe benötigte Funktion bzw. Methode aus der Dokumentation herausgesucht, bedarf es auch keiner weiteren Anmerkung. Direkt kopierter oder nur leicht modifizierter Code aus einer externen Ressource sollte im Skript jedoch entsprechend mit einem Kommentar gekennzeichnet werden. Dieser Kommentar sollte den Hinweis „Übernommen von“ oder „Basiert auf“ und die vollständige Literaturangabe, also Autor, URL und Zugriffsdatum enthalten. Dies kann z. B. folgendermaßen aussehen:

```
"""
Basiert auf der Antwort von Petr Viktorin auf die Frage „How do I create multiline comments in
Python?“: https://stackoverflow.com/a/7696966 (Zugriffsdatum: 6. Juli 2022)
"""
```

Programmcode aus dem Unterricht muss nicht gekennzeichnet werden.

3. Abgabe des Skripts

Das Programmierskript sollte stets in digitaler Form mit der Hausarbeit zusammen abgegeben werden. Daneben sollten auch alle zusätzlichen, für die fehlerfreie Ausführung des Skripts benötigten Dateien mit abgegeben werden, bspw. die verwendeten Daten(sets). Die verwendeten Daten sollten auch dann separat gespeichert und beigefügt werden, wenn der Zugriff und Download direkt im Skript, z. B. per Webscraping, erfolgt, um Problemen bei der Ausführung (Download-Barrieren, fehlendes Internet etc.) vorzubeugen.

Für die Abgabe gibt es mehrere Möglichkeiten:

- Per Mail, dafür alle Dateien in der entsprechenden Ordnerstruktur belassen und als ZIP-Archiv speichern und senden (Achtung: über die Uni-Mail können Python-Dateien nicht direkt versendet werden, deswegen müssen auch einzelne Dateien gezippt werden)
- Auf einem USB-Stick, der der Arbeit beigelegt wird;
- Über ein Repository, z.B. GitHub, Gitlab, Zenodo, Figshare. In diesem Fall muss in der Arbeit ein Link zu diesem Repository eingefügt sein.

Zusätzlich kann (kein Muss) das eigene Skript auch in den Anhang der Arbeit eingefügt werden. Der Anhang gehört hinter das Literaturverzeichnis und wird mit Seitenzahlen versehen. Besteht das Skript aus mehreren Teilen oder Einzeldokumenten, sollte jedes eigenständige Skript-Teil eine eigene Beschriftung erhalten (Anhang A, B, C, ...). Zum Einfügen des Codes kann analog zur Vorgehensweise beim Blockzitat vorgegangen werden, zudem sollten Zeilennummern analog zum Original ergänzt werden.

4. Mögliche Quellen für Daten(sets)

Für die meisten Hausarbeiten werden Daten(sets) als Grundlage benötigt, manchmal bietet es sich auch an, das Thema der Hausarbeit an den gefundenen Daten auszurichten, wenn diese bspw. in einem bestimmten (Annotations-)Format vorliegen müssen. Die folgende Liste an Datenrepositorien bietet daher ein paar Anlaufpunkte für die Suche nach geeigneten Daten:

- Kaggle
- Zenodo
- datahub
- World Bank Open Data
- SNAP
- figshare
- Million Song Dataset
- Informationisbeautiful

Eine ausführliche, thematisch sortierte Liste von Datensets, die sich für den Bereich des maschinellen Lernens eignen, ist auf Wikipedia zu finden. Darüber hinaus listet das w3c verschiede Datensets auf ihrer Seite auf. Eine weitere Möglichkeit für insbesondere XML- und Metadaten bieten digitale Sammlungen und digitale Editionen. Ist keine API oder ein (zentraler) Download mehrerer Dateien verfügbar, kann man auch die Herausgeber:innen kontaktieren.
Daneben bietet Google eine Dataset Search an. Auch Webscraping kann mit Python realisiert werden, alternativ können ganze Tabellen von Websites z. B. mit Excel extrahiert werden.

Eine weitere Möglichkeit der gezielten Datenabfrage bieten APIs, die von vielen Plattformen angeboten werden. Für Daten aus den sozialen Medien stellt bspw. die Twitter-API eine Möglichkeit dar, für Geo-Daten bietet OpenStreetMap die Overpass API an. Auf GitHub gibt es zudem ein API-Projekt, das Informationen zu öffentlichen APIs gesammelt zur Verfügung stellt.