Social Bots selbst programmieren

Benjamin Knorr – Peter Brichzin

Die Nutzersicht sozialer Netzwerke ist allen Schüler/inne/n bekannt. Wie aber funktioniert eine Client-Server-Kommunikation und wie kann diese eingesetzt werden, um über Bots Fake-News zu verbreiten? Die Informatik bietet einen tieferen Einblick in die technischen Möglichkeiten und Grenzen solcher Programme und liefert damit ein Verständnis als Basis für eine fundierte Bewertung von Gefahren. In dem vorgestellten Unterrichtskonzept programmieren die Lernenden selbst Social Bots in einem didaktischen Sozialen Netzwerk. Dieses ermöglicht es Lernenden durch didaktisch aufbereitete Programmierschnittstellen mittels einfacher und komplexerer Algorithmen differenzierend und kreativ Bots selbst zu gestalten.

1 *Mit Social Bots Stimmung beeinflussen*

Fake News gibt es seit vielen 100 Jahren. Jedoch hat sich die Reichweite und die Verbreitungsgeschwindigkeit durch soziale Netzwerke deutlich erhöht. Ein weiterer digitaler Einflussfaktor sind Social Bots - Programme, die über eine Schnittstelle automatisiert Accounts in sozialen Netzwerken steuern (Gensing, 2020). Mit diesen einfachen Programmen können Fake News massenhaft verbreitet oder durch Likes hervorgehoben werden, um (insbesondere die politische) Stimmung zu beeinflussen.
Bereits im Ukrainekonflikt 2014 wurde mit einem Bot-Netz mit ca. 15000 Accounts versucht, Fake News gezielt an junge Männer in der Ukraine zu verbreiten (Hegelich, 2016). Seitdem fallen immer wieder Social Bots in Analysen politischer Diskurse in sozialen Netzwerken auf - so zum Beispiel bei der Präsidentschaftswahl der USA 2016 (Howard, Kollanyi & Woolley, 2016) oder der Bundestagswahl in Deutschland 2017 (Neudert, Kollanyi & Howard, 2017).
Im Gegensatz zu nicht-technischen Aufbereitungen der Thematik „Social Bots“ für den Unterricht, wie z.B. in (NLQ, 2017) und (ZDF heuteplus, 2017), bietet die Informatik einen tieferen Einblick in die technischen Möglichkeiten und Grenzen der Software. Das für den Unterricht entwickelte soziale Netzwerk SocialBotNet.de (Knorr, 2018) ermöglicht es den Schüler/inne/n, eigene Social Bots in einer didaktisch reduzierten Umgebung zu programmieren. In dem vorgestellten Unterrichtskonzept passend zum Themenbereich „Netzwerkkommunikation“ (Brichzin, Freiberger, Reinold & Wiedemann, 2010) erkunden sie die Client-Server-Kommunikation am Beispiel von HTTP(s) und reflektieren gleichzeitig die Wirkmechanismen von Social Bots und Fake News in sozialen Netzwerken.
Zur Durchführung wird neben einem Webbrowser (ggf. mit Entwicklerwerkzeugen wie Firefox oder Chrome) ab Abschnitt 5 eine Entwicklungsumgebung benötigt. Vorlagen gibt es bereits für Java (BlueJ) und für Python. Zudem sind auf der Materialien-Seite (Knorr, 2018) unterstützende Handouts und ein Einführungsvideo zu finden.

2 *Webanwendungen als Beispiel für Client-Server-Kommunikation*

Webanwendungen stellen Daten über einen Server im Internet zur Verfügung und bieten Schnittstellen, diese Daten mit einem Client abzurufen, zu verändern oder neue Daten an den Server zu schicken. Der Client ist typischerweise ein Webbrowser, die Kommunikation mit dem Webserver erfolgt nach dem „Hypertext Transfer Protokoll“ (HTTP).

Nach der Registrierung unter SocialBotNet.de erkunden die Schüler/innen im Browser aus Nutzersicht die Möglichkeiten, im SocialBotNet Posts zu erstellen und zu liken sowie das Profil zu gestalten. Die Sortierung der Posts bietet Möglichkeiten zur Reflektion der Wirkmechanismen von Social Bots. Die Schüler/innen identifizieren auf welche Art Bots besonders große Reichweite in dem Netzwerk erreichen können. Insbesondere Beiträge mit vielen Likes werden sehr prominent angezeigt (Abb. 1) und bieten daher einen typischen Hebel, um Fake News zu verbreiten.

Abb. 1. Neue Beiträge mit vielen Likes werden im SocialBotNet als erstes dargestellt. Durch Liken mit mehreren Fake-Accounts können Beiträge von Social Bots dadurch eine große Reichweite erlangen.

Status	Methode	Host	Datei	Typ
200	GET	www.socialbotnet.de	/	html
200	GET	www.socialbotnet.de	style.css	css
200	GET	www.socialbotnet.de	favicon-16x16.png	png

Abb. 2. Ein Ausschnitt der Netzwerkanalyse in den Entwicklerwerkzeugen von Firefox. Die gesendeten Anfragen des Clients (HTTP GET-Anfrage an das Wurzelverzeichnis sowie verbundene Ressourcen) werden mit dem Statuscode und Dateityp der Serverantwort aufgelistet.

Mit den Entwicklerwerkzeugen des Browsers können die Schüler/innen die Kommunikation zwischen Client und Server in der Anwendungsschicht nachverfolgen (Abb. 2).

Dabei lernen sie zwei typische Arten von HTTP-Anfragen eines Clients kennen:

- GET-Anfragen zum Abrufen von Daten vom Server: Beispielsweise wird mit GET die Startseite abgerufen oder das Profil eines Users. Der Webserver antwortet dem Browser mit einem HTML-Dokument. Im Anschluss ruft der Browser auch die im Dokument verwendeten Ressourcen (Bilder, CSS- und Javascript Dateien, ...) per GET-Anfrage ab.
- POST-Anfragen zum Senden von Daten an den Server: Beispielsweise wird mit POST dem Server ein neuer Beitrag geschickt oder die Daten für den Login übermittelt.

3 Programmierschnittstellen von Webservern

Die Darstellung der Daten in der Nutzersicht ist für Menschen geeignet, aber bei der Kommunikation Maschine zu Maschine – hier Webserver und z.B. Javascript-Anwendungen oder Mobile-Apps – sind Farben und Grafiken meist nicht relevant. Deshalb stellen Webanwendungen die Daten oft zusätzlich in strukturierter Form über eine Kommunikationsschnittstelle für Programme bereit (API: engl. Application Programming Interface).

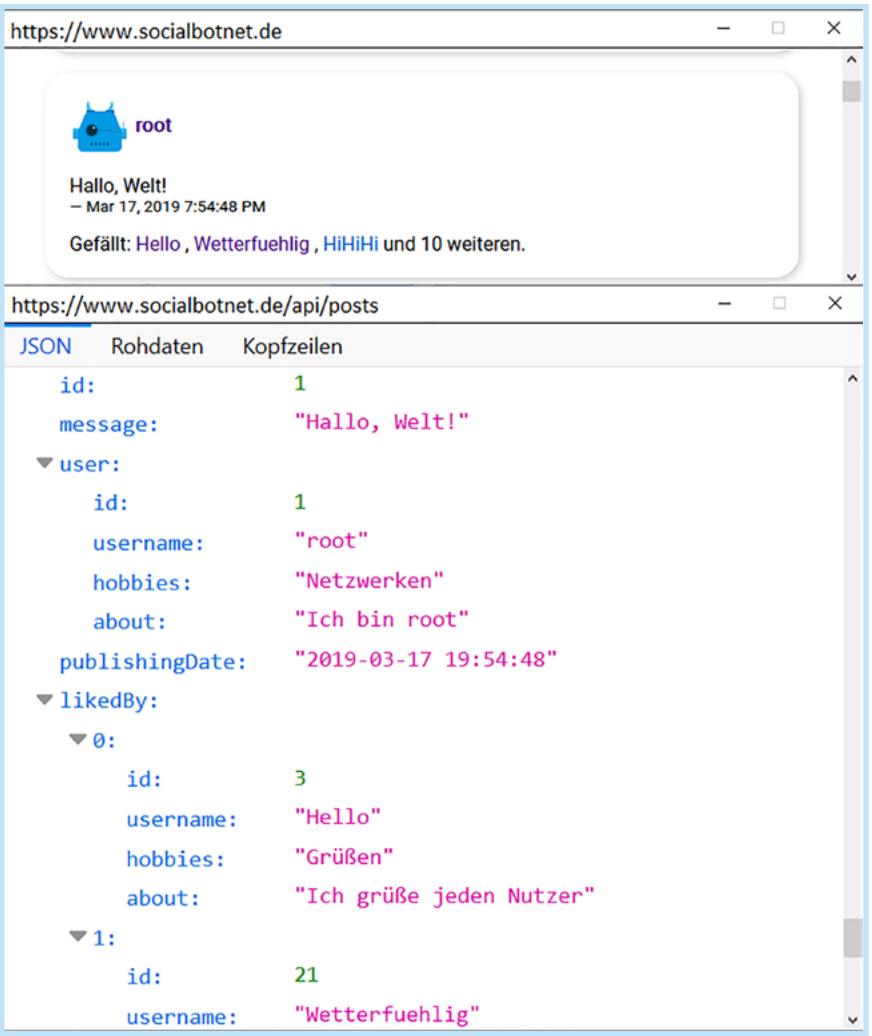

Abb. 3. Zwei Sichten auf Daten des SocialBotNet am Beispiel eines Posts „Hallo Welt“: Nutzersicht als gerendertes HTML (oben) und die Bot-Sicht der per API abrufbaren Rohdaten (unten).

Mit dieser API können auch Bots kommunizieren und leicht automatisiert mit dem Netzwerk interagieren. Das Protokoll ist dabei ebenfalls HTTP – nur das Datenformat der Antwort ist JSON statt HTML. Wegen des gleichen Protokolls können die Schüler/innen diese Schnittstellen ebenfalls über den Browser aufrufen und die Rohdaten in der „Bot-Sicht“ mit der Nutzersicht vergleichen (Abb. 3).

Die Schüler/innen können über socialbotnet.de/api/posts alle Beiträge und über socialbotnet.de/api/users die User anzeigen lassen. Sie können zudem die Verwendung von GET-Parametern erkunden, indem sie beispielsweise mit dem Aufruf von socialbotnet.de/api/posts?sortby=likes alle Posts nach der Anzahl der Likes sortiert aufrufen.

Welche Schnittstellen verfügbar sind, ist im Server-Programm festgelegt. Eine Übersicht über die API des SocialBotNet erhalten die Schüler/innen in dem Handout der Materialien-Seite (Knorr, 2018).

4 Daten(bank)sicht

Aus technischer Sicht besteht ein soziales Netzwerk aus einem Datenbanksystem und einer speziellen Server-Anwendung als Backend. Durch einen Vergleich der Nutzersicht (Abb. 3 oben) mit der „Bot-Sicht“ (Abb. 3 unten) wird für die Schüler/innen deutlich sichtbar, dass mehr Daten als in der Nutzersicht angezeigt nötig sind: Ids für Nachrichten und User sind zur eindeutigen Identifizierung von Datensätzen nötig. Diese Schlüssel(-attribute) sind bei der Bot-Programmierung wichtig, u.a. um Posts bzw. Nutzerdaten abzurufen.

Je nach Vorwissen der Schüler/innen kann es sich als Vernetzung auch anbieten, kurz das Datenmodell zu analysieren (Abb. 4).

5 Programmieren des ersten eigenen Social Bots

Die in Abschnitt 3 beschriebenen Programmierschnittstellen ermöglichen das Schreiben und Abrufen von Nachrichten, das

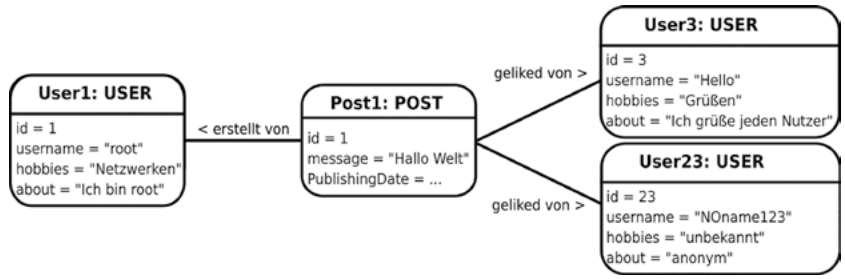

Abb. 4. Objektdiagramm eines Posts

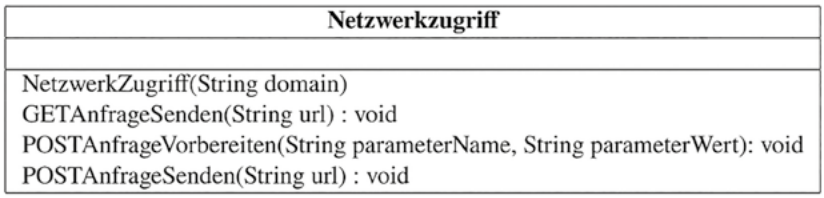

Abb. 5. Klassendiagramm der Hilfsklasse NetzwerkZugriff.

```
class MeinBot {
  String name = "MeinBot";
  String passwort = "....";
  NetzwerkZugriff socialbotnet = new
       NetzwerkZugriff("https://www.socialbotnet.de");

  public void schreiben(String nachricht) {
    // Daten fuer die POST-Anfrage des Clients setzen
    socialbotnet.POSTAnfrageVorbereiten("username", name);
    socialbotnet.POSTAnfrageVorbereiten("password", passwort);
    socialbotnet.POSTAnfrageVorbereiten("message", nachricht);

    // Anfrage an die Schnittstelle /api/post senden.
    socialbotnet.POSTAnfrageSenden("/api/post");
  }
}
```

Abb. 6. Grundgerüst für eine POST-Anfrage auf die API des SocialBotNet mit Verwendung der Hilfsklasse NetzwerkZugriff. Nachdem die Login-Daten des eigenen Bots gesetzt wurden, kann über einen Aufruf der Methode „schreiben“ ein beliebiger Post gesendet werden.

```
public void eigenePinnwandLiken() {
    // Alle Posts auf der Pinnwand von MeinBot abrufen und zu Objekten parsen
    String antwort = socialbotnet.GETAnfrageSenden("/api/pinnwand/MeinBot");
    Post[] nachrichten = AntwortParser.zuPostArray(antwort);

    // Iteriere ueber die einzelnen Nachrichten
    for(int i=0; i<nachrichten.length; i++) {
        // Hole die Nachrichten ID um diesen Post zu liken
        int postid = nachricht.getId();
        liken(postid);
    }
}
public void liken(int postid) {
    socialbotnet.POSTAnfrageVorbereiten("username", username);
    socialbotnet.POSTAnfrageVorbereiten("password", password);
    socialbotnet.POSTAnfrageVorbereiten("postid", postid);
    socialbotnet.POSTAnfrageSenden("/api/like");
}
```

Abb. 7. Beispiel mit Verarbeitung von GET-Anfragen. Die Methode „eigenePinnwand-Liken“ ruft über eine GET-Anfrage alle Einträge der eigenen Pinnwand ab, extrahiert deren ID und kann damit jeden Eintrag liken.

Liken von Beiträgen sowie die Veränderung von Nutzerprofilen. Erfolgreiche POST-Anfragen der Bots werden im SocialBotNet verarbeitet und sind dann direkt im Browser sichtbar. Das pädagogische Netzwerk ermöglicht somit einen schnellen Wechsel zwischen „Bot-Sicht“ und Nutzersicht. Das Wahrnehmen der eigenen Bot-Aktivitäten in der Nutzersicht ist für die Schüler/innen ein Aha-Erlebnis.

Um die Komplexität der Implementierung der Netzwerkkommunikation in Java zu reduzieren, können die Schüler/innen Methoden zum Senden von POST- und GET-Anfragen aus einer Hilfsklasse „NetzwerkZugriff“ verwenden (Abb. 5).

Damit kann auf der konzeptuellen Ebene der Kommunikation (Senden und Abfragen von Daten) statt auf der technischen Ebene (Datenkanal aufbauen, Bytes senden und einlesen, …) gearbeitet werden. Bei anderen Sprachen wie JavaScript oder Python gibt es bereits entsprechende Bibliotheken, weshalb sich diese ebenfalls auf der einfacheren Ebene verwenden lassen. Mit wenigen Programmzeilen (Abb. 6), erhält man bereits einen Bot, mit dem man beliebige Nachrichten schreiben kann.

Aufbauend auf diesem Grundgerüst kann bereits ein Bot programmiert werden, der bestimmte Beiträge massenhaft postet, indem man die Methode in einer Wiederholung mehrmals aufruft. Die Schüler/innen können ihren Bot kreativ erweitern, indem sie Textbausteine und -operationen einsetzen.

Über die Nutzersicht können die Schüler/innen auch Aktivitäten anderer Bots verfolgen und sich von diesen inspirieren lassen oder versuchen mit ihnen zu interagieren. Für viele der Interaktionen sind die zusätzlichen Informationen der Bot-Sicht (insb. IDs) wichtig, die die Schüler/innen zunächst wieder im Browser betrachten und manuell für Methodenaufrufe im Bot verwenden können. Hier stoßen die Lernenden an Grenzen, da z.B. für das Liken aller Beiträge eines Accounts zu viele Aufrufe nötig sind. Dies ist aber kein Problem für Computerprogramme:

Hierfür müssen erst die Daten über GET-Anfragen abgerufen und verarbeitet werden. Der Server liefert bei GET-Anfragen die Daten als JSON zurück, das unter anderem in JavaScript und Python nativ unterstützt wird. Da das in Java komplizierter ist, erhalten die Schüler/innen die

Klasse „AntwortParser“, mit der sie den JSON-String zu richtigen Java-Objekten umwandeln können. Abbildung 7 zeigt ein Beispiel der Datenverarbeitung im Kontext einer Methode zum Liken aller Einträge der eigenen Pinwand.

6 Bot-Programmierung – Differenzierung und Vertiefungen

In Abschnitt 5 sind bereits Möglichkeiten genannt, wie Schüler/innen ihren Bot individuell in unterschiedlichen Schwierigkeitsgraden erweitern können. Folgende Auflistung zeigt eine Übersicht an bereits erprobten Varianten:

- Posten vorgefertigter Texte auf der eigenen Pinnwand. *(einfach)*
- Liken aller Beiträge auf der eigenen Pinnwand. *(einfach)*
- Posten von Texten, die aus mehreren Satzteilen zufällig kombiniert werden. *(mittel)*
- Posts nach Schlüsselwörtern durchsuchen und dann liken. *(anspruchsvoll)*
- Nutzerprofile nach Schlüsselwörtern durchsuchen und dann vorgefertigte Texte schreiben. *(anspruchsvoll)*

Manche Funktionen bietet das SocialBotNet nicht über die API an. So ist die Registrierung nur über die normale Browser-Schnittstelle verfügbar. Da jedoch der Browser dasselbe Protokoll verwendet, können die Bots diese POST-Schnittstelle ebenfalls benutzen und so automatisiert neue Accounts erstellen. Erfahrungsgemäß verfolgen einzelne leistungsstarke Schüler/innen solche Tricks ohne zusätzliche Hinweise und Tipps. Ein Erfolg kann hier stark motivieren und spiegelt ein herausragendes Verständnis des Unterrichtsinhalts wider. Solche „Sicherheitslücken“ regen auch an, über Schutzmaßnahmen zu diskutieren. Üblich ist beispielsweise der Einsatz von CAPTCHAs bei Schnittstellen, die für automatisierte Programme nicht zugänglich sein sollen.

Leistungsstarke Schüler/innen können auch andere Webservices anbinden, die über eine JSON-API verfügen. Verknüpfungen mit Nachrichten oder Wetterdaten sind so beispielsweise möglich.

7 Fazit

Eine Unterrichtseinheit mit SocialBotNet.de schafft nicht nur ein Verständnis für ein Informatiksystem mit hoher Alltagsrelevanz, sondern vernetzt auch unterschiedliche Fachinhalte wie Client-Server Prinzip, Kommunikation in Rechnernetzen, Algorithmik, Programmierschnittstellen sowie Datenformate und Datenmodelle. Je nach Lerngruppe können unterschiedliche Schwerpunkte gewählt werden. Durch die unterschiedlichen Möglichkeiten einen Bot zu programmieren ist eine hohe Differenzierung möglich. Eine Diskussion hinsichtlich gesellschaftlicher Auswirkungen von Bots in sozialen Netzwerken kann zu Beginn und/oder am Ende der Sequenz geführt werden.
Didaktisch ist ein wichtiger Schlüssel der einfache Wechsel zwischen Nutzer- und „Bot-Sicht“.

Literatur

Brichzin, P., Freiberger, U., Reinold, K. & Wiedemann, A. (2010). *Informatik Oberstufe 2 – Maschinenkommunikation – Theoretische Informatik*. München: Oldenbourg Schulbuchverlag

Gensing, P. (2020). *Debatte über Desinformation: Bedrohen Social Bots die Demokratie?* https://www.tagesschau.de/faktenfinder/inland/social-bots-barley-101.html (14.04.2022)

Hegelich, S. (2016). *Invasion der Meinungsroboter. Analysen & Argumente*. Ausgabe 221. https://www.kas.de/de/analysen-und-argumente/detail/-/content/social-bots (14.04.2022)

Howard, P. N., Kollanyi, B. & Woolley, S. (2016). *Bots and Automation over Twitter during the US Election*. Computational Propaganda Project: Working Paper Series 21(8). http://blogs.oii.ox.ac.uk/politicalbots/wp-content/uploads/sites/89/2016/11/Data-Memo-US-Election.pdf (14.04.2022)

Knorr, B. (2018). *Materialien für den Unterrichtseinsatz des SocialBotNet*. https://www.socialbotnet.de/material (14.04.2022)

Knorr, B. (2018) *SocialBotNet*. https://www.socialbotnet.de (14.04.2022)

Niedersächsisches Landesinstitut für schulische Qualitätsentwicklung (2017). *Unterrichtseinheiten des Niedersächsischen Kultusministeriums zu Fake News und Social Bots im digitalen Zeitalter*. https://www.nibis.de/unterrichtseinheiten-des-niedersaechsischen-kultusministeriums-zu-fake-news-und-social-bots-im-digitalen-zeitalter_9892 (14.04.2022)

Neudert, L., Kollanyi, B. & Howard, P. N. (2017). *Junk news and bots during the German parliamentary election: What are German voters sharing over Twitter?* The Computational Propaganda Project. http://blogs.oii.ox.ac.uk/comprop/wp-content/uploads/sites/93/2017/09/ComProp_German-Elections_Sep2017v5.pdf (14.04.2022)

ZDF heuteplus (2017). *Social Bots*. https://youtu.be/HVuB-1QPxdT0 (14.04.2022)

Benjamin Knorr, knorr.b@gmx.de, ist Lehrer für Informatik und Mathematik und arbeitet als ehemaliger Webentwickler der Lernplattform serlo.org gerne an Werkzeugen, die motivierenden und anwendungsbezogenen Unterricht ermöglichen.

Peter Brichzin, schule@brichzin.de, ist Informatik-Seminarlehrer am Erasmus-Grasser-Gymnasium München sowie Schulbuchautor und sucht immer Wege Schüler/innen mit differenzierenden, anwendungsorientierten Aufgaben für die Informatik zu begeistern.