Raphael Zender et al. (Hrsg.): Die 18. Fachtagung Bildungstechnologien (DELFI),
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 235

Digitale Arbeitsblätter mit interaktiven Programmieraufgaben im Informatik-Unterricht

Sebastian Serth[1], Ralf Teusner[1] und Christoph Meinel[1]

Abstract: Moderner Informatikunterricht umfasst das Erlernen von Grundlagen des Programmierens. Dabei verwenden Lehrer häufig bereits vorhandene Videos, Quizfragen und praktische Programmieraufgaben aus Massive Open Online Courses (MOOCs), obwohl derzeit die Möglichkeiten zur Anpassung der Inhalte und dem Hinzufügen eigener Materialien für Lehrer begrenzt sind. Unsere Software ermöglicht es Lehrern, ihre eigenen interaktiven Arbeitsblätter mit angepassten und eigenen Übungen zu erstellen. Im Rahmen einer praktischen Evaluierung wurde das Konzept von Schülern und Lehrern gleichermaßen gut angenommen: Lehrer hatten mehr Zeit für die Beantwortung individueller Fragen und Schüler konnten in ihrem eigenen Tempo mithilfe automatisierter Rückmeldungen lernen. Für die Vorbereitung zukünftiger Unterrichtsstunden schätzten Lehrer die Möglichkeit, häufige Fehler auszuwerten, um so zuvor unerkannte Probleme besprechen zu können. Interaktive Arbeitsblätter fördern individualisierte Lernprozesse, unterstützen Lehrer in der Unterrichtsgestaltung und sind somit ein wichtiger Bestandteil digitaler Bildung an Schulen.

Keywords: Digitale Arbeitsblätter, praktische Programmieraufgaben, Informatik-Unterricht, Schule, MOOC

1 Einleitung

In modernem Schulunterricht kombinieren Lehrkräfte unterschiedliche Lernformen und verwenden dazu meist Schulbücher mit Erklärungen und Aufgaben oder stellen eigene Materialien mithilfe von Arbeitsblättern zusammen. In diesem Beitrag konzentrieren wir uns auf die Verwendung von Arbeitsblättern. Einer der größten Vorteile von Arbeitsblättern besteht darin, dass Lehrer die verwendeten Inhalte frei an ihre Bedürfnisse anpassen können. Die vorbereiteten Arbeitsblätter erhalten Schüler dann typischerweise in Papierform. Diese haben allerdings im Vergleich zu digitalen Varianten einige Nachteile, denn gedruckten Arbeitsblättern fehlt jegliche Form der Interaktivität.

Für Informatik-Kurse, insbesondere für solche, in denen Programmierkenntnisse vermittelt werden, ergeben sich zusätzliche Nachteile. Während eine Abbildung, z. B. eine schematische Darstellung eines Herzens im Biologieunterricht, von den Schülern leicht auf Papier annotiert werden kann, ist es schwierig, mit Quellcode auf Papier zu arbeiten. Insbesondere, wenn Schüler den Code ausführen oder verändern müssen, stellt diese Darbietungsform eine unnötige Hürde dar. Ein weiteres Problem tritt auf, wenn Lehrer beschließen, die Programme der Schüler einzusammeln, da diese typischerweise lokal auf dem jeweiligen PC gespeichert sind und die Lehrer somit die Mithilfe der Schüler benötigen.

[1] Hasso-Plattner-Institut, Universität Potsdam, Prof.-Dr.-Helmert-Str. 2-3, 14482 Postdam,
sebastian.serth@hpi.de, ralf.teusner@hpi.de, christoph.meinel@hpi.de

Ein möglicher Lösungsansatz für die angesprochenen Probleme besteht im Einsatz einer Cloud-Lösung für die Schulen. Eine solche ist dabei die HPI Schul-Cloud [Me19], die spezifisch für das deutsche Bildungssystem entwickelt wird. Derzeit fehlt dem Angebot jedoch eine dedizierte Unterstützung für den Informatikunterricht.

Angesichts der Fülle an Inhalten, die in Massive Open Online Courses (MOOCs) verfügbar sind, nutzen einige Lehrer die angebotenen Ressourcen mit ihren Schülern. Die Online-Kurse enthalten dabei typischerweise Videos und bieten Multiple-Choice-Quizze zur Festigung des Verständnisses oder interaktive Programmieraufgaben an. Die MOOC-Plattformen sind jedoch aktuell nicht auf die Anwendung in der Schule zugeschnitten. Daraus ergeben sich zwei wesentliche Nachteile: (1) Das Kursmaterial ist für Lehrer technisch nicht editierbar, sodass dieses nicht an die Bedürfnisse der Schüler angepasst werden kann und (2) die MOOC-Plattformen genehmigen Lehrkräften in der Regel nicht, Einblick in den Lernfortschritt ihrer Schüler nehmen.

Um diese Nachteile anzugehen, adressieren wir die folgende Forschungsfragestellungen:

1. Wie können wir Lehrer dazu befähigen, vorhandene Aufgaben (z. B. aus MOOCs) wiederzuverwenden, anzupassen und damit eigene, interaktive Arbeitsblätter zu erstellen?
2. Welchen Software-Support benötigen Lehrer, um Schüler zu unterstützen, die Probleme mit den gestellten Programmieraufgaben haben?
3. Wie können die gesammelten Ergebnisse zurück in den MOOC-Kontext fließen, damit eine größere Anzahl an Lernenden ebenfalls davon profitiert?

2 Stand der Forschung

Die folgenden Absätze geben einen kurzen Überblick über bisherige Forschungserkenntnisse, auf denen dieser Beitrag aufbaut. Dies umfasst insbesondere den Einsatz digitaler Inhalte im Unterricht und didaktische Ansätze zu Programmieraufgaben im Unterricht:

(1) Interaktive Inhalte im Unterricht: Othman et al. stellten 2013 fest, dass Informatikkurse zahlreiche abstrakte Konzepte enthalten, die sich nicht einfach mit traditionellen Lehrmethoden erklären lassen [OIP13]. Ein Vorteil interaktiver Arbeitsblätter wird von Blayney und Freeman beschrieben: Lernende werden in die Lage versetzt, in ihrer eigenen Geschwindigkeit zu lernen [BF08]. Damit wird die individuelle Gestaltung des Lernprozesses unterstützt, insbesondere dann, wenn ein interaktives Arbeitsblatt verwendet wird, das den Lernenden direktes Feedback gibt und so eine Hilfestellung genau dort bietet, wo Schüler die meiste Unterstützung benötigen. In Bezug auf den Grad der Interaktivität und des Feedbacks haben Blayney und Freeman in einer früheren Arbeit auch gezeigt, dass Multiple-Choice-Quizze nicht ausreichen und häufiges Feedback wichtig ist [BF04].

(2) Programmieraufgaben im Schulkontext: Bei dem Erlernen von Programmiergrundlagen besteht ein wichtiger Teil des Lernprozesses darin, das erworbene Wissen durch das

Schreiben von Quellcode (und dem Üben anderer Kompetenzen, wie z. B. der Modellierung) anzuwenden. Zusätzlich zur Verwendung einfacher Read-Eval-Print-Loops (REPL) wünschen sich Lehrer, dass ihre Schüler kleine Programme entwerfen. Ein guter Weg dazu wird von Isomöttönen et al. beschrieben, die vorschlagen, den Fokus der Schüler mit Quellcode als Vorlage auf bestimmte Themen zu lenken [ILL11]. Dies ist grundsätzlich auch durch Lösungen wie Jupyter (bspw. für Python) möglich, jedoch unterstützen diese die Entwicklung eigenständiger Programme in getrennten Dateien nur eingeschränkt. Plugins für Lernmanagementsysteme (LMS) wie CodeRunner [LH16] fokussieren sich wieder stärker auf die Abgabe einer Lösung und eigen sich nicht gut für deren Erarbeitung.

(3) Einsatz von MOOCs im Unterricht: Die meisten Lehrer, die MOOCs in ihren Unterricht integriert haben, waren mit den Ergebnissen zufrieden [Gr14]. Laut Griffiths et al. ist dies auf die gute Qualität der Inhalte und andere Lehrmethoden als in der Präsenzlehre zurückzuführen [Gr14]. M. Israel zeigte jedoch, dass die Anpassung bestehender Kurse, die nicht auf eine Nutzung in Schulklassen zugeschnitten sind, z. B. in Bezug auf das Engagement der Schüler und die Effektivität des Lernens eine große Herausforderung darstellt [Is15]. Zudem fanden Calfield et al. heraus, dass Lernende aufgetretene Fragen lieber mündlich als über eine Online-Community stellen [CCH13]. Auch Lehrer profitieren von mündlichen Diskussionen, da sie so leichter Einblick in den Fortschritt und Probleme ihrer Schüler bekommen. Dies ist insbesondere von Vorteil, da Lehrer im Allgemeinen keinen direkten Zugang zu den Aktivitäten ihrer Schüler in einem MOOC haben [Is15].

(4) Learning Analytics: Erkenntnisse über das Lernverhalten, so genannte Learning Analytics, können von Lehrern nicht nur zur Leistungsbewertung herangezogen werden, sondern auch für die Verbesserung ihres Unterrichts benutzt werden [MSW14]. Lehrer können die bereitgestellten Informationen zur Vorbereitung zukünftiger Unterrichtsstunden nutzen und so z. B. eine zielgerichtete Wiederholung einbauen, um bestehende Lücken im Verständnis eines Themas zu schließen. In Bezug auf Programmieraufgaben schreibt Blikstein, dass die Häufigkeit der Ausführungen und die Anzahl der Quellcode-Änderungen Rückschlüsse auf die Methodik erlaubt und so einen Einblick in die Herangehensweise und Denkansätze der Lernenden gewährt [Bi11]. Berland et al. betonen, dass oben beschriebene Ergebnisse entweder von einem Lehrer oder für automatisiertes Feedback an die Lernenden verwendet werden können [BBB14].

3 Herangehensweise

Im Rahmen dieses Beitrags haben wir qualitative Interviews mit 13 Informatiklehrern, zwei Schulleitern und fünf Schülern geführt. In diesen haben wir Erkenntnisse über häufig verwendete Unterrichtsmethoden gesammelt und mehr über die verfügbaren Ressourcen erfahren. Unsere Interviewpartner hoben dabei insbesondere die Möglichkeiten zur Wiederverwendung bestehender Materialien für die Erstellung von Arbeitsblättern hervor. Aus den Interviews haben wir folgenden Schwächen klassischer Arbeitsblätter unabhängig von der konkreten Verteilungsform (Papier / Word-Dokument / PDF) abgeleitet:

(1) Mangelnde Interaktivität: Einige Lehrer verwenden Beispiele aus Videos oder Online-Spielen wie Pac-Man, um Schüler zu motivieren, an einer bestimmten Aufgabe zu arbeiten (z. B. dem Erarbeiten eines Pfadfindungsalgorithmus für die Geister des Spiels Pac-Man). Ein vorbereiteter Programmrahmen ist dann jedoch von dem gezeigten Beispiel losgelöst und lässt sich durch die Schüler nicht direkt im Rahmen des Arbeitsblatts bearbeiten.

(2) Mangelnde Anpassungsfähigkeit: Andere Lehrer verwenden Inhalte aus MOOCs, z. B. aus den Programmierkursen zu Java und Python, die auf openHPI [MW13] verfügbar sind. In diesen Kursen wird eine browserbasierte Umgebung zum Programmieren und Ausführen von Quellcode namens *CodeOcean* verwendet [St16]. Die Plattform wird von Lehrern als geeignet eingestuft, sie vermissen jedoch eine Bearbeitungsfunktion für Aufgaben. Daher können die Aufgaben für ein Arbeitsblatt nur unverändert mittels Link übernommen werden oder Lehrer müssen einige der Vorteile von *CodeOcean* einbüßen.

(3) Mangelnde Benutzerfreundlichkeit: Durch die digitale Bearbeitung der Programmieraufgaben können Ergebnisse nicht durch Einsammeln von ausgefüllten Arbeitsblättern vollumfänglich gesammelt werden. Einige andere Lehrkräfte kritisierten die Nachteile klassischer Abgabesysteme, die auf dem manuellen Hoch- und Herunterladen von Dateien basieren, sodass diese für den Einsatz im Informatikunterricht impraktikabel erscheinen.

Durch die Analyse mit uns geteilter Arbeitsblätter konnten wir weitere Bedürfnisse ableiten. So verteilen einige Lehrer ihre Arbeitsblätter beispielsweise als bearbeitbare Textdokumente statt als PDF-Dateien, damit Schüler die Dokumente individualisiert ausfüllen können. Andere entscheiden sich für die Nutzung eines LMS, über das Schüler neben der Abgabe von Lösungen auch automatisiert auswertbare Multiple-Choice-Quizze beantworten können. Dennoch fehlt dabei eine Verknüpfung bereitgestellten Quellcodes und anderer Materialien, sodass beim Lernen immer wieder Kontextwechsel erforderlich sind. Wenn Lehrer sich zur Umgehung dieser Nachteile entschließen, z. B. durch die Verwendung von MOOC-Plattformen, müssen sie jedoch Einschränkungen bei der Bearbeitbarkeit der Inhalte und den fehlenden Zugriff auf Lösungen ihrer Schüler akzeptieren. Folglich gibt es derzeit keine Option, die den Bedürfnissen der Lehrer vollständig gerecht wird.

4 Konzept der interaktiven Arbeitsblätter

Unser Ziel ist es, Lehrer in die Lage zu versetzen, vorhandene Inhalte in Arbeitsblättern zu editieren und zu erweitern. Die größte Herausforderung für Lehrer stellt dabei die Einbindung und Anpassung von Programmieraufgaben aus MOOCs dar. Um Kontextwechsel zu reduzieren und die Analogie zu einem papierbasierten Arbeitsblatt zu erhöhen zeigen wir die Aufgaben direkt auf derselben Seite wie anderes Material an. Neben formatiertem Text unterstützt unser Arbeitsblatt-Editor auch Bilder und Videos sowie Multiple-Choice-Quizze. Zudem umfasst unser Konzept auch eine tiefgreifende Integration von Learning Analytics. Wie im vorigen Abschnitt erwähnt, stehen diese Daten den Lehrern normalerweise nicht zur Verfügung, obwohl insbesondere sie von speziell aufbereiteten Ansichten

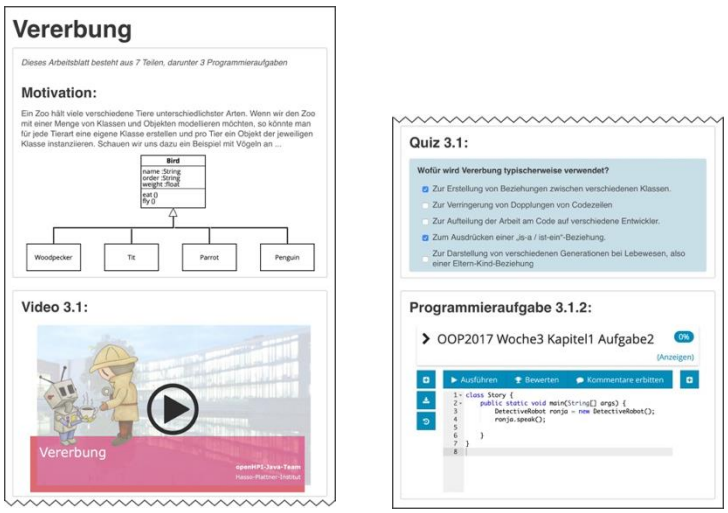

Abb. 1: Beispiel eines Arbeitsblatts zum Thema „Vererbung", wie wir es in unserer Evaluation verwendet haben. Das Arbeitsblatt besteht aus einer textuellen Erklärung, einem Bild, einem Video, einer Multiple-Choice-Frage und einer Programmieraufgabe.

auf die Daten profitieren könnten. Mit einer detaillierten Analyse jeder Programmieraufgabe soll Lehrern so Einblick darin gewährt werden, wie viel Zeit Schüler für die Bearbeitung einer Übung investiert haben und welche Kommentaranfragen gestellt wurden.

Das vorgeschlagene Konzept zielt darauf ab, Lehrern die Erstellung eigener Arbeitsblätter unter Verwendung anpassbarer Inhalte aus MOOCs zu ermöglichen (siehe Abb. 1). Beim Hinzufügen einer interaktiven Programmieraufgabe sollen Lehrer zudem die Möglichkeit haben, Optionen für die Einbindung einer Aufgabe anzugeben. So wollten manche Lehrkräfte das automatisierte Feedback der Unit-Tests deaktivieren oder die erzielte Punktzahl, z. B. im Rahmen einer Prüfung, ausblenden. Der Arbeitsblatt-Editor muss daher diese Einschränkungen unterstützen und sie beim Aufruf von Programmieraufgaben anwenden.

5 Prototypische Umsetzung

Aus technischer Sicht sind drei Komponenten an der Umsetzung unseres Konzepts beteiligt: (1) Die HPI Schul-Cloud als Identitätsanbieter, (2) ein Arbeitsblatteditor namens *edtr.io* und (3) die Ausführungsumgebung *CodeOcean*. Durch das Aufrufen eines Arbeitsblattes wird die Identität des jeweiligen Benutzers und dessen Rolle (Schüler/Lehrer) für eine berechtigungsabhängige Darstellung und das Speichern des Fortschritts verwendet.

edtr.io ist ein erweiterbarer, offener Rich-Content-Editor, der Benutzerdaten und Dokumenten in einer relationalen Datenbank speichert und auf diese über GraphQL zugreift. Durch Echtzeitaktualisierungen können Lehrer bestimmte Informationen während der Schulstunde ein- oder ausblenden und so die Aufmerksamkeit ihrer Schüler lenken.

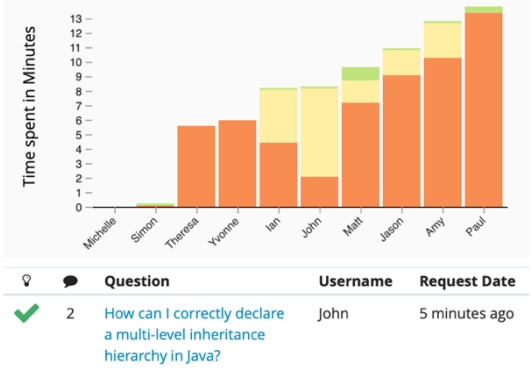

Abb. 2: Das Live-Dashboard stellt die Arbeitszeit dar und schlüsselt über die verschiedenen Farben den aktuellen Fortschritt der Schüler auf: orange (< 50%), gelb (50%-89%), grün (≥ 90%). Außerdem werden unten kürzlich gestellte Kommentaranfragen angezeigt.

Programmieraufgaben können mit wenig Aufwand sowohl umfänglich angepasst (Aufgabenstellung, Testfälle, Vorlage) als auch deren Einbindung samt verfügbarer Features (Code-Editor, Testergebnisse, Hilfestellung, Ansichtsmodus) individualisiert werden.

Beim Aufruf von Programmieraufgaben wird eine pseudonymisierte ID, die für jede Benutzer- und Service-Kombination einmalig ist, die jeweilige Rolle (Schüler/Lehrer) sowie eine Kurs-ID an *CodeOcean* übergeben. Damit werden automatisch Lerngruppen mit allen Mitgliedern der jeweiligen Klasse gebildet werden. In *CodeOcean* können Lehrer zudem detaillierte Informationen ihrer Lerngruppen über ein Dashboard erhalten, das die wichtigsten Aktivitäten pro Aufgabe zusammenfasst (siehe Abb. 2). Es zeigt die von den Schülern investierte Zeit in Korrelation mit den erzielten Punkten und listet Kommentaranfragen, die von den Schülern während der Arbeit an der Aufgabe gestellt wurden.

6 Evaluierung

Um sicherzustellen, dass unser Konzept und der entwickelte Prototyp den Bedürfnissen der Nutzer entspricht, haben wir in insgesamt 27 Interviews regelmäßig Feedback zu unterschiedlichen Entwicklungsständen eingeholt.

Um die Auswirkungen unseres Konzepts einschätzen zu können, ist es wichtig zu verstehen, wie Schüler und Lehrer unser Konzept bewerten und wie der Einsatz digitaler Arbeitsblätter den Unterricht verändert. Daher war es das Ziel unserer Evaluierung, möglichst früh Feedback aus typischen Unterrichtsstunden mit unserem Prototypen zu erhalten. Unsere Evaluierung führten wir mit einer Klasse durch, die bereits Java mit Hilfe eines Online-Kurses auf openHPI lernte, um so den generellen Einfluss der Online-Inhalte zu minimieren. In diesem Kurs folgten den Videos Multiple-Choice-Fragen und Programmieraufgaben in *CodeOcean*. Für einen ersten Testlauf bereiteten wir in enger Absprache mit dem Lehrer zwei Arbeitsblätter zu den Java-Themen „Methoden“ und „Vererbung“

vor (siehe Abb. 1). In einem weiteren Durchgang testeten wir ein Arbeitsblatt zu „abstrakten Klassen“, das zudem Referenzen auf das Schulbuch enthielt.

6.1 Methodik

Die entworfenen Arbeitsblätter wurden während einer regulären Schulstunde von einem gemischten Informatik-LK aus Baden-Württemberg mit Schülern der 11. und 12. Klasse verwendet. Um eine Beeinflussung unserer Untersuchung durch unsere Anwesenheit auszuschließen, entschieden wir uns gegen eine Hospitation des Unterrichts. Daher lag es in der Verantwortung des Lehrers, das Thema wie üblich vorzustellen und die Schüler durch das Material zu führen. Nach dem Unterricht wurden alle Teilnehmer gebeten, unsere anonyme Umfrage auszufüllen und somit Feedback zu ihren Erfahrungen zu geben. Darüber hinaus haben wir den Lehrer wieder interviewt, um mehr über die Situation in der Klasse und seine Einschätzung zu erfahren. In der Umfrage für die Schüler wollten wir wissen, wie ihnen das Konzept im Allgemeinen gefiel und welche Vor- oder Nachteile sie in dem Einsatz der verschiedenen Medien (d.h. einem Schulbuch, traditionellen Arbeitsblättern und unserem Ansatz) sehen. Zusätzlich erhoben wir den Net Promoter Score (NPS; wie wahrscheinlich es ist, dass sie interaktive Arbeitsblätter an Freunde weiterempfehlen würden), wie er von Reichheld beschrieben wird [Re03]. Zudem haben wir, zur Quantisierung des Nutzungserlebnisses, in unsere Umfrage Teile des Fragebogens zur „modularen Evaluation zentraler Aspekte der User Experience“ (meCUE) [Mi17] eingebunden.

Ferner stellten wir unseren Prototypen interessierten Lehrern auf der didacta, einer der größten europäischen Bildungsmessen, vor. Währenddessen sammelten wir weitergehende Ideen und baten die Lehrkräfte um ihr Feedback zum aktuellen Stand mit Hilfe des standardisierten User Experience Questionnaire (UEQ) [LHS08].

6.2 Ergebnisse

Nach der ersten Unterrichtsstunde mit unseren Arbeitsblättern berichtete der Lehrer, dass seine Schüler gut zurechtkamen. Er schätzte die tiefe Integration der verschiedenen Ressourcen sowie den Materialmix, und brachte wiederholt die Absicht zum Ausdruck, selbst gleichwertige Arbeitsblätter erstellen zu wollen. Seiner Beobachtung zufolge wechselten

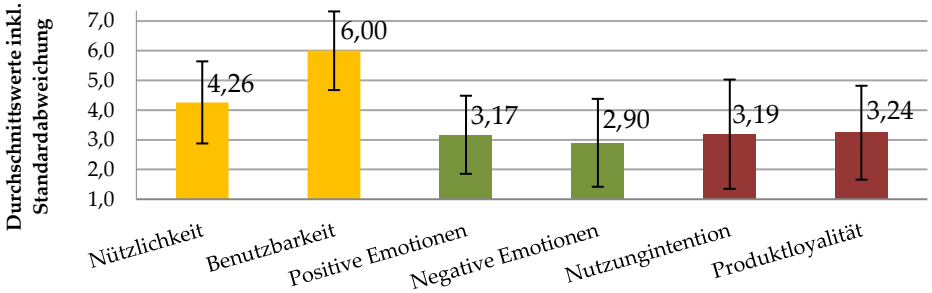

Abb. 3: Ergebnisse der Module I (gelb), III (grün) und IV (rot) des meCUE. (N=7)

die Schüler während der Arbeit an den Programmieraufgaben häufiger zu früheren Teilen der Arbeitsblätter im Vergleich zur bisherigen Lernform mit Online-Kursen.

Von den 21 Schülern der Klasse nahmen neun an unserer freiwilligen Umfrage teil, von denen nur wenige ihre Antworten direkt am Ende der Stunde gaben. Acht Schülerinnen und Schüler schätzten das allgemeine Konzept und das Gesamtdesign unserer interaktiven Arbeitsblätter als gut ein. Sie lobten zudem die gebotene Übersicht über Struktur und Inhalt der Unterrichtsstunde. Außerdem hoben sie die allgemeine Verfügbarkeit der Online-Inhalte (bspw. im Gegensatz zu dateibasierten Arbeitsblättern) hervor, da sie so unabhängig von ihrem jeweiligen Standort auf die Inhalte zugreifen konnten, z. B. auf dem Weg zur Schule. In unserer Umfrage erhielten wir einen NPS von −38 (der zwischen −100 und +100 liegen kann; N=8). Die Antworten auf den meCUE zeigten dagegen eine Gesamtpunktzahl von durchschnittlich 2,4 (Bereich von −5 bis +5; N=7). Weitere Werte des meCUE von „stimme stark zu“ = 7 bis „stimme stark nicht zu“ = 1 sind in Abb. 3 zu sehen.

Von 17 Lehrern, die unseren Prototyp ausprobierten, füllten zwölf den UEQ aus. Die durch den UEQ erfassten Skalen erwiesen sich meist als gut oder ausgezeichnet (vgl. Abb. 4), insbesondere wurde die Stimulation (Mittelwert 1,7) als am besten bewertet. Im Vergleich zum UEQ-Benchmark, der einen Vergleich mit über 400 anderen Studien erlaubt, hat die Durchschaubarkeit (1,4) das größte Verbesserungspotenzial.

6.3 Diskussion und Interpretation der Ergebnisse

Die Ergebnisse motivieren, den Einfluss interaktiver Arbeitsblätter weiter zu erforschen. Die Umfrageantworten lassen einige, zunächst vorläufige Schlussfolgerungen zu und deuten auf die gute Benutzerbarkeit des Konzepts hin (vgl. meCUE Modul I). Insbesondere die Antworten der Schüler zeigen dabei Widersprüche auf, wie den Unterschied zwischen dem NPS (−38) und dem meCUE-Gesamturteil (2,4). Auch die Freitextantworten zeigen kleinere Widersprüche, was darauf hindeutet, dass einige Teile unserer Umfrage falsch interpretiert worden sein könnten. Zudem könnte der negative NPS und die neutrale Emotion im meCUE Modul III durch bestätigte Verbindungsprobleme im Schulnetzwerk zum Testzeitpunkt erklärt werden, die auch den Zugriff auf die Umfrage beeinträchtigte. Nach Aussage des beteiligten Lehrers war das Gesamturteil der Schüler ansonsten eher positiv.

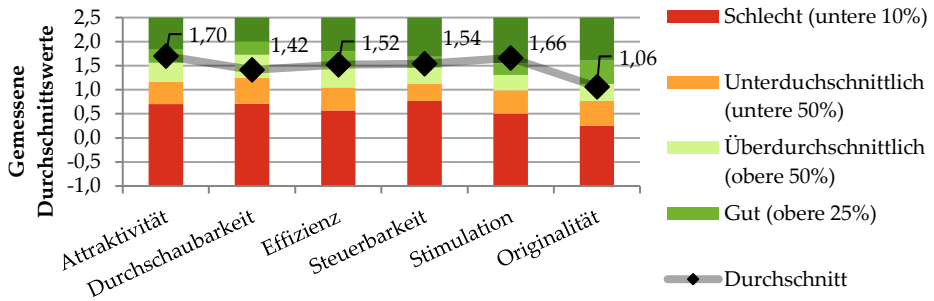

Abb. 4: Durchschnittswerte des UEQ im Vergleich zum Benchmark des Fragebogens. (N=12)

Aus den Textkommentaren der Schüler ziehen wir drei Schlussfolgerungen: (1) Beim Vergleich der interaktiven Arbeitsblätter mit MOOCs und separaten Arbeitsblättern heben sie die klarere Strukturierung und den verbesserten Überblick der interaktiven Arbeitsblätter hervor. (2) Sie schätzen die Mischung verschiedener Inhaltstypen auf einer einzigen Webseite und sehen keinen Vorteil darin, stattdessen die MOOC-Plattform aufzurufen. (3) Klassische Arbeitsblätter werden als unflexibler angesehen, diese dürfen allerdings manchmal als Nachschlagewerk in Klausuren verwendet werden. Insgesamt schätzen Schüler die Integration der Onlineinhalte und die Einführung moderner Lernmethoden.

Die größten Vorteile interaktiver Arbeitsblätter wurden von Lehrern geäußert. Sie mögen die Einfachheit, mit der individuelle Arbeitsblätter mit Programmieraufgaben erstellt werden können. Unsere qualitative Umfrage zeigt, dass das Konzept den Anforderungen der Informatiklehrer entspricht und die Ergebnisse des UEQ unterstreichen das vorgeschlagene Design. Basierend auf den Erkenntnissen der Learning Analytics gewinnen Lehrkräfte ein besseres Verständnis dafür, warum ihre Schüler mit Aufgaben Schwierigkeiten haben. Die angebotene Darstellung bietet dabei die benötigten Informationen auf einen Blick und wird durch die hinterlegten Unit-Tests automatisch generiert. Während des Unterrichts beobachteten die beteiligten Lehrkräfte, dass die Schüler in ihrem eigenen Tempo zwischen dem Lernen und Anwenden des Stoffs wechselten. Daher berichteten die Lehrer, dass interaktive Arbeitsblätter und das automatisierte Feedback ihnen dabei helfen, von einem Dozenten der Klasse zu einem Lernhelfer für individuelle Schüler zu werden. Das positive Feedback der Lehrer und deren wiederholte Fragen zur Erstellung eigner Arbeitsblätter unterstreicht die Relevanz und praktische Anwendbarkeit unseres Konzepts.

7 Ausblick und zukünftiger Forschungsbedarf

Identifizierte Probleme, wie die fehlende Verwendbarkeit der digitalen Arbeitsblätter in Prüfungssituation, sollten mit einer Funktion zum Ausdrucken bearbeiteter Arbeitsblätter behoben werden können. Damit verlieren die Schüler jedoch die Vorteile interaktiver Arbeitsblätter und können während papierbasierter Klausuren ebenfalls nicht auf die gewohnten Funktionen zugreifen. Zukünftige Forschungsarbeiten sollten sich daher auf die speziellen Anforderungen während einer Prüfung konzentrieren. In einem „Prüfungsmodus“ sollten Lehrer die verfügbaren Funktionen noch detaillierter steuern können, z. B. indem ein Zeitlimit vorgegeben wird und Abgaben explizit durchgeführt werden können.

Da die Einführung interaktiver Arbeitsblätter in hohem Maße davon abhängt, wie einfach es für Lehrer ist, eigene Arbeitsblätter zu erstellen, und wie viele Vorlagen bereits zur Verfügung stehen, möchten wir auch Möglichkeiten zum Teilen von Arbeitsblättern und zur Einbindung freier Bildungsressourcen (Open Education Resources, OER) untersuchen. Insbesondere die Erstellung der benötigten Unit-Tests zur automatischen Auswertung erfordert einige Mühen und limitiert Lehrer ansonsten möglicherweise in der Nutzung. Zudem möchten wir unsere Experimente mit einer größeren Nutzergruppe erneut durchführen, um zuverlässige und potenziell verallgemeinerbare Ergebnisse zu erzielen.

8 Zusammenfassung

Der Informatikunterricht an Schulen umfasst praktische Programmieraufgaben, durch die technische Anforderungen an Schulcomputer gestellt werden. Daher mögen Lehrer Online-Programmierumgebungen (z. B. als Bestandteil von MOOC-Plattformen), die ohne lokale Einrichtung genutzt werden können. Informatiklehrer schätzen die verfügbaren Online-Inhalte und verweisen daher auch von klassischen Arbeitsblättern darauf. Damit Lehrkräfte Inhalte an die jeweiligen Bedürfnisse ihrer Klasse anpassen können, entwickelten wir das Konzept der interaktiven Arbeitsblätter mit Programmieraufgaben. Auf Grundlage unserer Experimente können wir daher unsere Forschungsfragen wie folgt beantworten:

1. Wie können wir Lehrer dazu befähigen, vorhandene Aufgaben (z. B. aus MOOCs) wiederzuverwenden, anzupassen und damit eigene, interaktive Arbeitsblätter zu erstellen?

Lehrer benötigen Zugriff auf die gleichen Werkzeuge wie MOOC-Anbieter und profitieren daher von den neu geschaffenen Möglichkeit zur Aufgabenbearbeitung in *CodeOcean* sowie der Integration in *edtr.io*. Zudem profitieren sie von Möglichkeiten zum Austausch mit und der Weiterverwendung von Aufgaben ihrer Kollegen. In unserem Konzept können alle Übungen in ein interaktives Arbeitsblatt eingebunden und die den Schülern zur Verfügung stehenden Funktionen mithilfe einiger Parameter nach Belieben angepasst werden.

2. Welchen Software-Support benötigen Lehrer, um Schüler zu unterstützen, die Probleme mit den gestellten Programmieraufgaben haben?

Die Unterstützung durch Software verbessert die Interaktion zwischen Schülern und Lehrern in Bezug auf die folgenden drei Aspekte:

(1) Gesamtüberblick und Einblick in den Bearbeitungsfortschritt der Schüler: Zusätzlich zum Zugriff der Lehrer auf die Abgaben der Schüler bei Bedarf profitieren Lehrer von dem entwickelten Dashboard. Durch die dortige Visualisierung der Ergebnisse der Schüler können Lehrer ihre Schüler bei Verständnisschwierigkeiten gezielt unterstützen und kommende Unterrichtsstunden besser vorbereiten.

(2) Just-in-time-Feedback: Die Echtzeitverarbeitung der Lerndaten in unserem Tool empfanden Lehrer im Rahmen der Evaluierung von Lehrern als hilfreich, da so aufkommende Fragen beantwortet und mögliche Missverständnisse direkt erkannt werden konnten.

(3) Inhaltliche Fragen: Während einer Unterrichtsstunde wünschen Lehrer und Schüler keine technische Unterstützung bei ihrer Kommunikation, da sie die direkte, mündliche Kommunikation bei auftretenden Problemen schätzen. Für Hausaufgaben profitieren Schüler jedoch von der Möglichkeit, Fragen an ihre Lehrer zu stellen oder sich mit zeilenweisen Kommentaren in *CodeOcean* gegenseitig zu helfen.

3. Wie können die gesammelten Ergebnisse zurück in den MOOC-Kontext fließen, damit eine größere Anzahl an Lernenden ebenfalls davon profitiert?

Die verbesserte Struktur der Arbeitsblätter im Vergleich zu einem MOOC wurde von den Schülern geschätzt, und unsere Auswertung legt nahe, dass sie die Lernenden beim erneuten Zugriff auf das Lernmaterial während des Programmierens unterstützt. Zusätzlich zur Unterteilung in feste Wochen profitieren MOOCs zudem von einer besseren Gruppierung der Lernmaterialen desselben Themas. Dies verbessert ebenso die Navigation im Inhalt. Darüber hinaus legt unsere Forschung [Se19] nahe, dass das Konzept der Lerngruppen auf MOOCs angewendet werden sollte, damit ein Tutor die Lernenden individuell durch den Kurs führen kann. So könnten die Tutoren die Kontrolle über die Freischaltung zusätzlicher Inhalte oder die Anpassung der Abgabefristen bekommen und auch Zugang zu den Learning Analytic-Daten erhalten, um informierte Entscheidungen zu ermöglichen.

In diesem Beitrag haben wir digitale Arbeitsblätter mit interaktiven Programmieraufgaben vorgestellt und ausgewertet. Durch die Kombination von Text, Videos und interaktiven Inhalten, einschließlich praktischer Programmieraufgaben und Multiple-Choice-Fragen auf einem interaktiven Arbeitsblatt, verbessern wir den Status Quo sowohl für Schüler als auch für Lehrer: Die Schüler können in ihrem eigenen Tempo lernen und Lehrern wird der Wechsel vom Vortragenden zum persönlichen Tutor erleichtert. Zudem können Lehrer ihre Schüler durch weitere analytische Einblicke besser unterstützen. Die erhaltenen Rückmeldungen aus der Praxis sind überwiegend positiv und zeigen damit das enorme Potenzial, das die Einführung digitaler Inhalte im Unterricht bietet.

9 Danksagung und sprachlicher Hinweis

Wir bedanken uns bei allen Beteiligten der Studie für die engagierte Unterstützung. Zur besseren Lesbarkeit wird im Beitrag überwiegend das generische Maskulinum verwendet, diese Formulierung soll jedoch ausdrücklich Personen jeden Geschlechts umfassen.

Literaturverzeichnis

[BBB14] Berland, M.; Baker, R. S.; Blikstein, P.: Educational Data Mining and Learning Analytics: Applications to Constructionist Research. Technology, Knowledge and Learning Bd. 19, S. 205–220, 2014.

[BF04] Blayney, P.; Freeman, M.: Automated formative feedback and summative assessment using individualised spreadsheet assignments. Australasian Journal of Educational Technology Bd. 20, S. 209–231, 2004.

[BF08] Blayney, P.; Freeman, M.: Individualised interactive formative assessments to promote independent learning. Journal of Accounting Education Bd. 26, S. 155–165, 2008.

[Bi11] Blikstein, P.: Using learning analytics to assess students' behavior in open-ended programming tasks. In (Long, P. et. al., Hrsg.): Proceedings of the 1st International Conference on Learning Analytics and Knowledge - LAK '11. Banff, Canada, S. 110, 2011.

[CCH13] Caulfield, M.; Collier, A.; Halawa, S.: Rethinking Online Community in MOOCs Used for Blended Learning, 06.10.2013, https://er.educause.edu/articles/2013/10/rethinking-online-community-in-moocs-used-for-blended-learning, Stand: 03.04.2020.

[Gr14] Griffiths, R. et. al.: Interactive Online Learning on Campus: Testing MOOCs and Other Platforms in Hybrid Formats in the University System of Maryland. Ithaka S+R, New York, USA, 2014.

[ILL11] Isomöttönen, V.; Lakanen, A.-J.; Lappalainen, V.: K-12 game programming course concept using textual programming, In (Cortina, T. J. et. al., Hrsg.): ACM Technical Symposium on Computer science education – SIGCSE '11. Dallas, TX, USA, S. 459, 2011.

[Is15] Israel, M. J.: Effectiveness of Integrating MOOCs in Traditional Classrooms for Undergraduate Students. The International Review of Research in Open and Distributed Learning Bd. 16, S. 102–118, 2015.

[LHS08] Laugwitz, B.; Held, T.; Schrepp, M.: Construction and Evaluation of a User Experience Questionnaire. In (Holzinger, A., Hrsg.): HCI and Usability for Education and Work Bd. 5298. Springer Berlin Heidelberg, Berlin u.a., S. 63–76, 2008.

[LH16] Lobb, R.; Harlow, J.: Coderunner: A Tool for Assessing Computer Programming Skills. ACM Inroads Bd. 7 Nr. 1, S. 47–51, 2016.

[Me19] Meinel, C. et. al.: Die HPI Schul-Cloud: Roll-Out einer Cloud-Architektur für Schulen in Deutschland. Universitätsverlag Potsdam, Potsdam, 2019.

[Mi17] Minge, M. et. al.: The meCUE Questionnaire: A Modular Tool for Measuring User Experience. In (Soares, M.; Falcão, C.; Ahram, T. Z., Hrsg.): Advances in Ergonomics Modeling, Usability & Special Populations Bd. 486. Springer International Publishing, S. 115–128, 2017.

[MSW14] Monroy, C.; Snodgrass Rangel, V.; Whitaker, R.: A Strategy for Incorporating Learning Analytics into the Design and Evaluation of a K-12 Science Curriculum. Journal of Learning Analytics Bd. 1, S. 94–125, 2014.

[MW13] Meinel, C.; Willems, C.: openHPI: the MOOC offer at Hasso Plattner Institute. Universitätsverlag Potsdam, Potsdam, 2013.

[OIP13] Othman, A.; Impes, A.; Pislaru, C.: Online Interactive Module for Teaching a Computer Programming Course. In (Ciussi, M.; Augier, M., Hrsgb.): Proceedings of the 12th European Conference on e-Learning ECEL 2013. 2013.

[Re03] Reichheld, F. F.: The One Number You Need to Grow. Harvard Business Review, 2003.

[Se19] Serth, S. et. al.: Evaluating Digital Worksheets with Interactive Programming Exercises for K-12 Education. In 2019 IEEE Frontiers in Education (FIE). Cincinatti, USA, 2019.

[St16] Staubitz, T. et. al.: CodeOcean - A versatile platform for practical programming excercises in online environments. In 2016 IEEE Global Engineering Education Conference (EDUCON). Abu Dhabi, S. 314–323, 2016.