Synthesis of Social Media Messages and Tweets as Feedback Medium in Introductory Programming

Full Paper
SACLA 2019

Sonny Kabaso[1[0000−0001−6550−1445]] and Abejide Ade-Ibijola[2[0000−0001−9507−0455]]

Formal Structures, Algorithms, and Industrial Applications Research Cluster
Department of Applied Information Systems
School of Consumer Intelligence and Information Systems
University of Johannesburg, Auckland Park Bunting Road Campus
Johannesburg, South Africa
smulenga.k@gmail.com, abejideai@uj.ac.za, www.abejide.org

Abstract. Social Media has been recognised as a supportive tool in Education, creating benefits that supplement student collaboration, class interactions and communication between instructors and students. Active informal interactions and feedback between instructors and students outside class is one of the main reasons behind Social Media pedagogy. With many innovative usage methods of Social Media in Education this creates new opportunities, one being automatic feedback for students. Despite the prevalence of traditional email methods of providing feedback to students, many studies show that they do not check their emails as frequent as they check their Social Media accounts. In this paper, we present the automatic generation of feedback messages and tweets using a Context-Free Grammars (CFG). Our design takes a class list of students and their mark sheets and automatically composes tweets (using the CFG rules) about statistical "fun facts" about programming problems, exercises, class performances, and private messages about individual student performances. These tweets and messages are then pushed to Tweeter using the Twitter Application Programming Interface (API). A survey of 116 student participants at a South African university showed that majority of the students will love to get such notifications on Social Media, rather than check their emails; and that Lecturers also find this initiative to be a forward thinking one.

Keywords: Synthesis of Things, Social Media, Tweet Synthesis, Context-Free Grammar, Introductory Programming, Procedural Generation.

1 Introduction

Social Media is collective of web-based applications built off social dogma and the foundations of Web 2.0 technology consisting of a multitude of user generated

and shared information [1]. Over the years Social Media has become one of the most popular internet services in the world [2]. It has made it possible for communication for many people [3], and has become an essential part of our everyday lives [4].

Despite the fact that Social Media was only used for social communication when it was first introduced, it can be used in a professional context and has evolved into a platform for student engagement and interaction [5, 6]. In Education, Social Media has created a centered platform for student interaction, collaboration and feedback [7, 8]. For instance, platforms like Twitter has driven instant and active interaction between instructors and their students, student participation, formal communication outside the classroom and continued learning when students are in their own spaces [9–11]. Digital natives are frequent Social Media users and would rather receive information using modern technology, this aids traditional learning and teaching methods [8, 12, 13].

Some studies have shown that when social platforms like Twitter are used in an educational setting, learning is more engaging and stimulating than traditional teaching methods [11, 14]. Students also prefer to use mobile devices for Social Media because they have access to their devices at any time, by this extending the time of interaction with them outside the classroom [5, 8, 15]. Thus incorporating Social Media in Education drastically improves the amount of time students interact with their studies and this is beneficial for new students in introductory programming.

Introducing new students to programming requires time and patience [16, 17], it is considered a challenge by many computer programming instructors [18]. Many novice programmers find learning new programming concepts difficult, with various publications providing evidence of high failure and drop-out rate in programming courses [19, 16, 20–22]. Novice programmers have minimal engagement with programming concepts and problems they fail to create clear mental models of programme execution [23]. In order to address this, we need to develop teaching methods that enable regular practice by programming students, preferably using social media as a tool [24]. Given that many institutions still depend on traditional pedagogical models of teaching in the digital era, this limits the amount of time students have with course content as they are always on Social Media [6]. Hence, it has become imperative that we create more innovative methods of engaging with the digital natives in programming classrooms is desirable.

One innovation will be to send push notifications to novice programmers on Social Media, automatically. With Social Media's popularity, creating supplementary communication methods among students and instructors can contribute to positive learning [15].

In this work, we have presented the synthesis of Social Media messages and tweets as a feedback medium for novice programmers by employing techniques in Natural Language Generation (NLG), using formal grammars. This involves generating tweets from a collection of processed data [25, 26]. We have achieved this by employing Context-Free Grammars (CFG) in the generation of short unique

messages as tweets from student data. Each of these tweets are categorised into broadcasts or direct messages. Broadcasts include performance stats, learning guides, test announcements, exercises, assignments, new topics and concepts; whilst direct messages include: individual student attendance and performance. This processes is described in Figure 1.

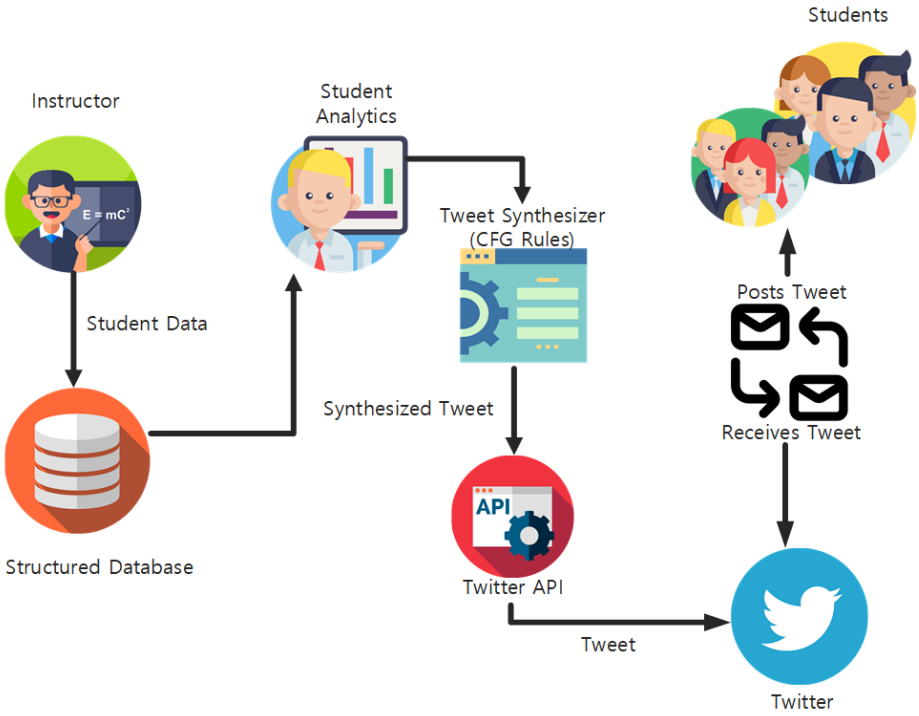


Fig. 1. Processes of tweet feedback generation using CFG's

In Figure 1, an instructor inserts a class list of students data into a database. This data this then analysed to determine their performance and attendance. The tweet synthesizer then automatically generates feedback tweets and messages. These tweets and messages are then pushed to Twitter using an API[1].

The following are the contributions made in this paper. We have:

1. developed a method of automatically generating educational feedback to support Introductory Programming in the form of tweets using CFG rules,
2. built a tool called the `Tweet Synthesiser` that automatically generates and posts programming concepts, practice exercises and solutions in the form of tweets for novice programmers, and

[1] Application Programming Interface

3. evaluated how useful the `Tweet Synthesiser` is in an educational setting, especially for novice programmers.

The rest of this paper presents the following. Section 2 outlines the background and related works. Section 3 describes CFG for tweet generation. Section 4 shows the implementation of the suggested tool and real examples of the generated tweets. Section 5 evaluates the idea and Section 6 defines a conclusion including further work to be done.

2 Background and Related Work

This Section introduces the problem, discussing the related work and motivation behind this work.

2.1 Research Questions

This paper addresses the following research questions of interest to solve the identified problem.

1. Can we generate personalised Social media messages and tweets as feedback based on real time data on student attendance and performance with limited or no lecturer involvement?
2. Can we improve learning methods for Introductory Programming and make it more interactive outside normal classes in an informal context?
3. How can we design CFG rules to answer the above questions?

2.2 Motivation

Here we present challenges faced by programming instructors and students.

Location and Time Constraints: Time and location constraints become a major setback in a traditional setting because students only interact with course work during school time [8].

Student Preference: Students use Social Media more compared to email [8, 13]. A comparison between the use of Social Media and email technologies indicated that students preferred to use the Social Media technologies rather than email [15].

Course Difficulty: Programming is a difficult subject to learn as a newbie [19, 16, 20, 21]. In order to obtain any long-term knowledge and skill it is important that novice programmers have continuous interaction and practice with programming problems [24].

2.3 Why Twitter?

In this Section, we describe why we have chosen Twitter as the Social Media Platform to push programming feedback for Introductory Programming. Twitter is a popular micro-blogging platform for social networking which was established in 2006. It is estimated to have 554.7 million active users around the world [27], a number of them being digital natives and students [5]. It allows for people to communicate using short (280 character limit) text messages or status updates known as tweets. These tweets can either be posted using the Twitter application, instant/text messaging including other third party applications like Facebook, email and other websites using the Twitter API [28]. Twitter is easily accessible regardless of the geographical location of the user.

Twitter being a push technology, feedback is instant because the Twitter application is readily available on students mobile devices [8]. A study by Buettner 2012 created a curriculum to push concepts, subject-related resources and new topics to Twitter with each student having direct notifications on the mobile devices every morning. The majority of the students in this study found this practice much more engaging and useful [29, 14].

2.4 Structure of a Tweet

A tweet can contain up to 280 characters or less including spaces [30]. In this paper we describe a tweet to be comprised of four components which include:

1. **text:** the short message that a user wants to deliver and targets the context of a specified audience,
2. **hashtag:** is a word or phrase preceded with the hash (#) symbol. This can indicate a topic, event or community associated to the tweet. A tweet can consist of more that one hashtag inclusive to the 280 character count,
3. **mention:** is the username or Twitter handle of a specific user appended with the "at" (@) symbol, and
4. **url:** is a web link to an external source outside the Twitter application. This is usually related to the context of the tweet.

All components of a tweet are optional but at least one component must be present in the tweet.

2.5 Related Work and Terms

This Section lists some previous works in Natural Language Generation using similar technologies.

Natural Language Generation: Generation of Social Media profiles using probabilistic CFG's on Facebook [31], human-like language generation using Monte Carlo Tree Search employing context-free grammars [32], sentences generation for probabilistic TAG grammars [33, 34], generating narratives of SQL queries using context-free grammars [35] and in-game text generator using expressive free text markup and CFG's [36].

Problem Generation: Automatic generation of python practice problems using context-free grammars [37] and automatic generation of regular expression practice problems and solutions [38].

A term that appears many times in this paper is Context-Free Grammar. Here, we present the definition of this term, as follows:

Definition 1 (Context-free Grammar (CFG) [39]). *A CFG is a quadruple* $G = (N, \Sigma, P, S)$*, where:*

1. N *is a restricted set of elements known as non-terminal or syntactic variables.*
2. Σ *is a restricted alphabet of terminal symbols where* $\Sigma \not\subseteq N$*. Terminals appear on the right side of Replacing Rules*
3. P *is a set of Replacing Rules, with the form* $A \longrightarrow a$*, where* $A \in N$ *and* $a \in (N \cup \Sigma)^*$*.*
4. S *is the selected starting non-terminal where* $S \in N$*.*

3 Grammar Design for Tweet Synthesis

Here we present the CFG rules for synthesising tweets as a grammar $G = (N, \Sigma, P, S)$, with the rules defined, starting from building blocks, in Section 3.1.

3.1 Building Block

In Rule 1 we define the `<identifier>` as using the regular expression $[A-Za-z0-9\_]^+$ in BNF[2] notation. Rule 2 states that a mention must be preceded by the @ symbol representing the tweeter handle of the user where m `<mention>` $\in$ [@`<identifier>`]$^+$, Similarly in Rule 3 where `<hashtag>` h should always be appended with the # symbol. In Rule 5 we define the message while Rule 4 defines the context of the message being tweeted. Replacing Rule 8 defines any url represented as u which must always be less than 280 characters.

$$\texttt{<identifier>} \longrightarrow [A-Za-z0-9\_]^+ \tag{1}$$

$$\begin{aligned} \texttt{<mention>} &\longrightarrow m \in [@\texttt{<identifier>}]^+ \\ &\longrightarrow \ni: m \in (N \cup \Sigma) \end{aligned} \tag{2}$$

$$\begin{aligned} \texttt{<hashtag>} &\longrightarrow h \in [\#\texttt{<identifier>}]^+ \mid \lambda \\ &\longrightarrow \ni: h \in \Sigma^+ \end{aligned} \tag{3}$$

$$\begin{aligned} \texttt{<context>} &\longrightarrow \texttt{<exercise>} \mid \texttt{<schedule>} \mid \texttt{<attendance>} \mid \\ &\longrightarrow \texttt{<performance>} \end{aligned} \tag{4}$$

$$\texttt{<msg>} \longrightarrow \texttt{<context><url>} \mid \texttt{<context>} \mid \texttt{<url>} \tag{5}$$

$$\texttt{<optional\_txt>} \longrightarrow \texttt{<text>} \mid \lambda \tag{6}$$

$$\tag{7}$$

[2] Backus-Naur Form

$$\texttt{<url>} \longrightarrow [\ (http(s)?://)?([\backslash w-]+\backslash.)+[\backslash w-]+[.com|co.za|.org|.net]+(/[/?\backslash\%\&=]*)?\ ] \ \wedge \ \ni: \mid \texttt{<url>} \mid \ \leq \ 280 \tag{8}$$

3.2 Tweet

A tweet can either be a broadcast `<broadcast_tweet>` or a direct message `<inbox_tweet>`. `<broadcast_tweet>` is only limited to 280 characters inclusive of all components in Rule 9.

$$\begin{aligned} \texttt{<tweet>} &\longrightarrow \texttt{<broadcast\_tweet>} \mid \texttt{<inbox\_tweet>} \\ &\ni: \mid \texttt{<broadcast\_tweet>} \mid \ \leq \ 280 \end{aligned} \tag{9}$$

3.3 Broadcast Tweet

Contrary to what we state in Sub-section 2.4, our context in rule 10 requres that the `<msg>` component must always be present while the `<mention>` and `<hashtag>` remain optional.

$$\begin{aligned} \texttt{<broadcast\_tweet>} &\longrightarrow \texttt{<msg>} \mid \texttt{<msg><hashtag>} \mid \\ &\longrightarrow (\texttt{<mention><msg>}\mid\texttt{<msg><mention>})\texttt{<hashtag>} \mid \\ &\longrightarrow \texttt{<hashtag>}(\texttt{<mention><msg>}\mid\texttt{<msg><mention>}) \mid \\ &\longrightarrow \texttt{<msg>}(\texttt{<mention><hashtag>}\mid\texttt{<hashtag><mention>}) \mid \\ &\longrightarrow \texttt{<mention>}(\texttt{<msg><hashtag>}\mid\texttt{<hashtag><msg>}) \end{aligned} \tag{10}$$

To derive a `<broadcast_tweet>` for an exercise and solution we can further break-down the variables for `<exercise>` and `<solution>` in Rules 11 and 12. `<optional_txt>` in rule 6 can be any other wording that makes sense to the context of the tweet.

$$\begin{aligned} \texttt{<exercise>} &\longrightarrow \texttt{<optional\_txt><category><topic>} \\ &\longrightarrow \texttt{<optional\_txt><problem\_no><problem\_txt>} \\ &\longrightarrow \texttt{<optional\_txt>} \end{aligned} \tag{11}$$

$$\begin{aligned} \texttt{<solution>} &\longrightarrow \texttt{<optional\_txt><category><topic>} \\ &\longrightarrow \texttt{<optional\_txt><solution\_no><solution\_txt>} \\ &\longrightarrow \texttt{<optional\_txt>} \end{aligned} \tag{12}$$

$$\begin{aligned} \texttt{<schedule>} &\longrightarrow (\texttt{<event>} \mid \texttt{<optional\_txt>}) \mid \\ &\longrightarrow (\texttt{<optional\_txt>} \mid \texttt{<event>}) \mid \texttt{<schedule>} \end{aligned} \tag{13}$$

With a schedule tweet we define the `<event>` from the `<schedule>` variable in Rule 13.

3.4 Inbox Tweet

Here we describe replacment rules for deriving a direct inbox message tweet. In rule 14 <tag> is one or more hashtags.

$$\texttt{<tag>} \longrightarrow (\texttt{<hashtag>} \mid \lambda)^{+} \tag{14}$$

$$\texttt{<inbox\_tweet>} \longrightarrow \texttt{<msg><tag>} \mid \texttt{<tag><msg>} \tag{15}$$

$$\texttt{<performance>} \longrightarrow \texttt{<optional\_txt><marks><module><optional\_txt><event>} \tag{16}$$

$$\texttt{<attendance>} \longrightarrow \texttt{<optional\_txt><avg\_attendance><optional\_txt>} \tag{17}$$

Unlike a <broadcast_tweet> which is limited to 280 characters or less, an inbox tweet can consist of more than 280 characters. Some examples of tweets derivations are shown below.

Example 1 (Derivation of a Direct Message). In this example <inbox_tweet> we derive an example of a direct message about a student's recent performance on a class test from Replacing Rule 15.

$$\texttt{<inbox\_tweet>} \Longrightarrow \texttt{<msg><hashtag>} \tag{18}$$

$$\Longrightarrow \texttt{<performance><hashtag>} \tag{19}$$

$$\Longrightarrow \texttt{<optional\_txt><marks><module><optional\_txt><event><hashtag>} \tag{20}$$

$$\Longrightarrow \textit{You Scored } \texttt{<marks><module><optional\_txt><event><hashtag>} \tag{21}$$

$$\Longrightarrow \textit{You Scored } 50\% \texttt{ <module><optional\_txt><event><hashtag>} \tag{22}$$

$$\Longrightarrow \textit{You Scored } 50\% \textit{ DSW1A}\ \texttt{<optional\_txt><event><hashtag>} \tag{23}$$

$$\Longrightarrow \textit{You Scored } 50\% \textit{ DSW1A}\ \textit{on } \texttt{<event><hashtag>} \tag{24}$$

$$\Longrightarrow \textit{You Scored } 50\% \textit{ DSW1A}\ \textit{on TEST1 } 28/06/2019\ \texttt{<hashtag>} \tag{25}$$

$$\Longrightarrow \textit{You Scored } 50\% \textit{ DSW1A}\ \textit{on TEST1 } 28/06/2019\ \textit{DSW1A\_ASSESSMENTS} \tag{26}$$

Example 2 (Schedule tweet). This example shows a derivation of a broadcast tweet announcement for a schedule class test from rule 10

$$\texttt{<broadcast\_tweet>} \Longrightarrow \texttt{<msg><hashtag>} \tag{27}$$

$$\Longrightarrow \texttt{<event><optional\_txt><hashtag>} \tag{28}$$

$\Longrightarrow$ <optional_txt><event>
<optional_txt><hashtag> (29)

$\Longrightarrow$ *You have* <event> <optional_txt>
<hashtag> (30)

$\Longrightarrow$ *You have Homework scheduled for*
28/06/2019 <optional_txt><hashtag> (31)

$\Longrightarrow$ *You have TEST*1 *scheduled for*
28/06/2019 *Dont forget to add it to your*
calender<hashtag> (32)

$\Longrightarrow$ *You have TEST*1 *scheduled for*
28/06/2019. *Dont forget to add it to your*
*calender#DSW*1*A_ASSESSMENTS* (33)

Example 3 (Derivation of an Exercise Tweet). In this example we show a derivation of an exercise broadcast tweet on for-loops from rule 10.

<broadcast_tweet> $\Longrightarrow$ @*student* <msg><hashtag> (34)

$\Longrightarrow$ @*student* <optional_txt><category><topic>
<hashtag> <optional_txt><problem_no>
<problem_txt> <optional_txt><hashtag> (35)

$\Longrightarrow$ @*student check out this* <category> *exercise* :
<problem_no> <problem_txt>. *This problem is on*
<topic> *a solution is tweeted with*<hashtag>. (36)

$\Longrightarrow$ @*student check out this Python Programs exercise*
: <problem_no> <problem_txt>. *This problem is on*
<topic> *a solution is tweeted with*<hashtag>. (37)

$\Longrightarrow$ @*student check out this Python Programs*
exercise : 1 <problem_txt>. *This problem is on*
<topic> *a solution is tweeted with*<hashtag>. (38)

$\Longrightarrow$ @*student check out this Python Programs*
exercise : 1 *Write a piece of code* (*using a FOR loop*)
to display all odd numbers between 110 *and* 152.
This problem is on <topic> *a solution* (39)

$\Longrightarrow$ @*student check out this Python Programs*
exercise : 1 *Write a piece of code* (*using a FOR loop*)
to display all odd numbers between 110 *and* 152.
This problem is on LOOPS a solution
is tweeted with it <hashtag>. (40)

$$\begin{aligned}
&\Longrightarrow \text{@}student\ check\ out\ this\ Python\ Programs \\
&exercise\ :\ 1\ Write\ a\ piece\ of\ code\ (using\ a\ FOR\ loop) \\
&to\ display\ all\ odd\ numbers\ between\ 110\ and\ 152. \\
&This\ problem\ is\ on\ LOOPS\ a\ solution \\
&is\ tweeted\ with\ it\ \#SolutionToProblem1.
\end{aligned} \tag{41}$$

4 Implementation

In Section 3, we presented the rules for generating tweets. In this Section, we have implemented these rules in a push notification tool for messages and feedback on Twitter. This tool, developed using the .Net framework, is shown in Figure 2. The tool is divided into four sections for each category of the tweet categories namely *exercises/solutions*, *student performance*, *attendance* and *schedule*. With an exercise tweet, we first select a programming category and topic for the desired tweets. For instance, the category would be *algorithm design* and the topic *if-statements*. With those selections a problem exercise and solution is automatically generated using AAI API [37]. This generates procedural practice algorithms and programs in Python.

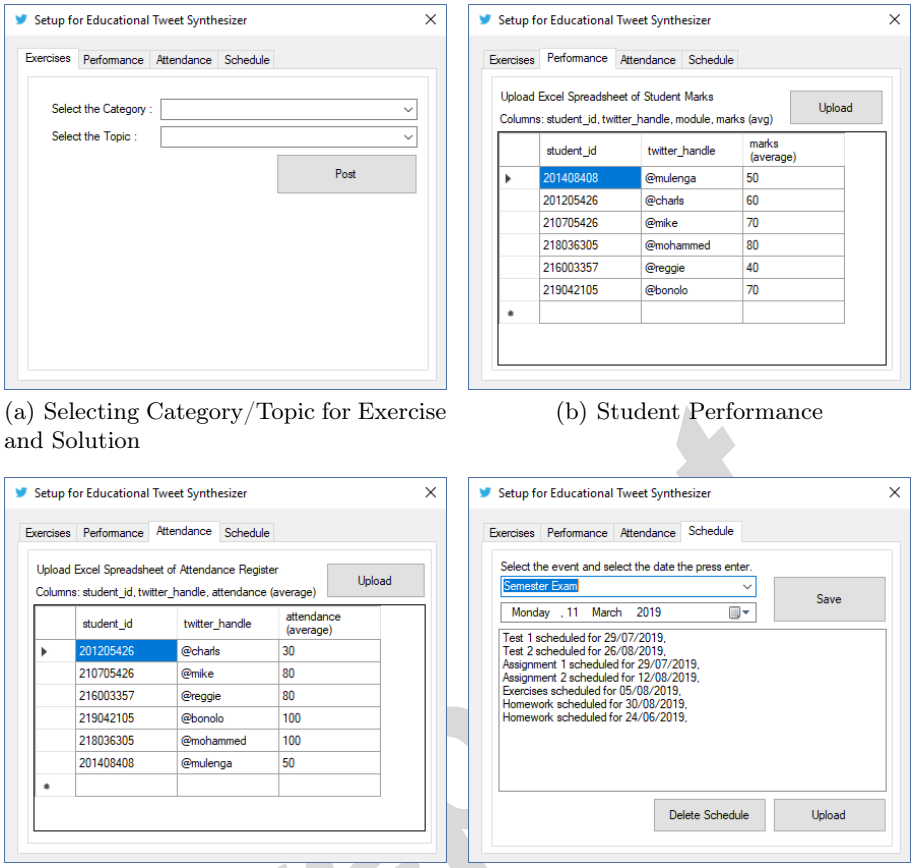
The problem and solution are then passed through the tweet synthesizer to post a broadcast exercise tweet with a problem and solution at an interval. Figure 4 is a representation of the generated tweets for a problem exercise and solution. The `Tweet Synthesiser` generated 500 tweet exercises and 500 tweet solutions for their corresponding questions in a 1 minutes of execution. A five second interval is placed between the posting of each of tweet to avoid flooding the Twitter API.

Performance and attendance tweets work similarly, we select a class list with the following fields (student_id, twitter_handle, module, and marks/attendance) as a CSV file. With that data, we composed custom tweet messages depending on their performance average and class attendance, that are sent directly to students. We generate schedule/deadline tweets from a digital study-guide by adding important events or dates. Figure 3 represents the flow of tweets generation using the developed tool.

5 Evaluation

5.1 Students Perception of Tweet Synthesiser

This Section presents the results from a survey evaluation on students opinions on receiving module comments/announcements on Social Media using the `Tweet Synthesiser`. The online survey was conducted at the University of Johannesburg in South Africa. The majority of the respondents were enrolled in Information Technology and Information Systems related degrees taking programming courses.



(a) Selecting Category/Topic for Exercise and Solution

(b) Student Performance

(c) Attendance Register

(d) Schedule and Deadlines

Fig. 2. Setup for Educational `Tweet Synthesiser`

A total of 116 responses was received. About 92% of the respondents were students between the age of 18 to 25, 6.2% were above the age of 25 while the remaining 1.8% were less than 18 years of age in Figure 5(a). Students were asked if their instructors used Social Media to support teaching and learning, only 17.7% claimed *Yes* considering platforms like WhatsApp as Social Media while 82.3% stated *No* (see Figure 5(b)).

Students were asked if they frequently used Social Media, most of them said to be frequent Social Media users with 81.3% ranging from a scale of 6 to 10 and 19% not being that frequent with 10 being very frequent and 1 not using Social Media at all. 76.1% claimed to be frequent on Facebook, 66.4% Instagram, 44.2% Twitter users, with some students using all three of the above mentioned social platforms in Figure 5(e). They were also asked how frequent they access their student email in Figure 5(c), compared to how frequent they use Social Media only 8.8% claimed to regularly check their emails with 43.4% claiming to check

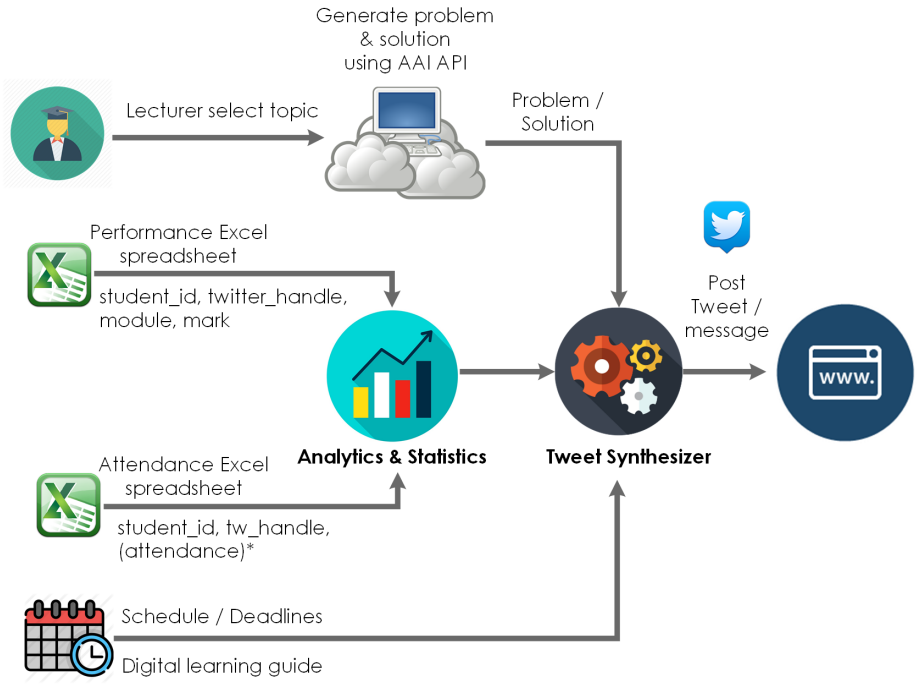


Fig. 3. Implementation of `Tweet Synthesiser`

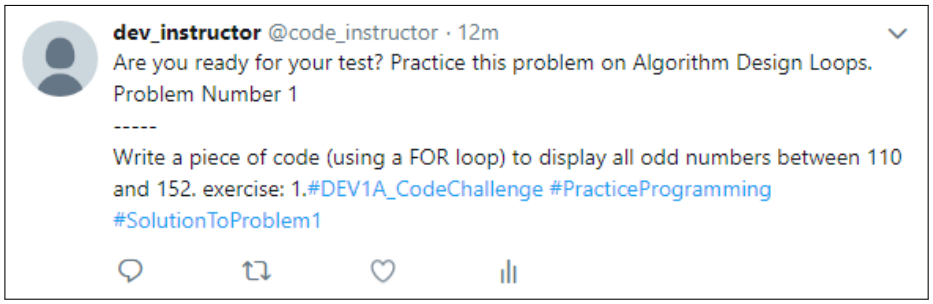
their emails 2 to 5 times a day while the rest would check only once a day or rarely at all.

When the students were asked if they preferred class feedback and learning on Social Media, 86.7% of them agreed to receiving feedback on Social Media. Part of that majority preferred that the tool be implemented on other social platforms suggesting Facebook, Instagram and Whatsapp allowing them to continuously be exposed to programming concepts and regularly practice in Figure 5(e). With this we conclude that automating course feedback on Social Media would be beneficial for continues learning outside classrooms.

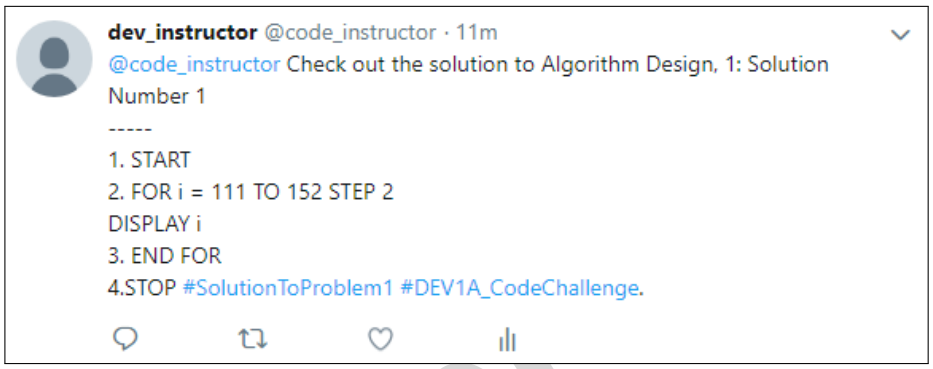
6 Conclusion and Future work

6.1 Conclusion

In this paper, we have presented a CFG for the automatic generation of Social Media messages and tweets for novice programmers. We have demonstrated a method for automating feedback in Education on Social Media. Other than Twitter this technique can be used on other popular social platforms such as Facebook and Instagram. Our evaluation also shows that a majority of students in our survey are frequent Social Media users and would prefer this form of feedback on other social platforms.



(a) Sample Synthesised Problem Tweet



(b) Sample Synthesised Solution Tweet for (a)

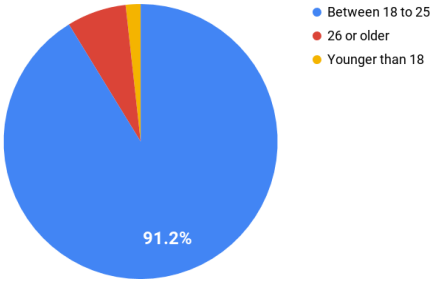
Fig. 4. Sample Synthesised Exercise Tweet

6.2 Future Work

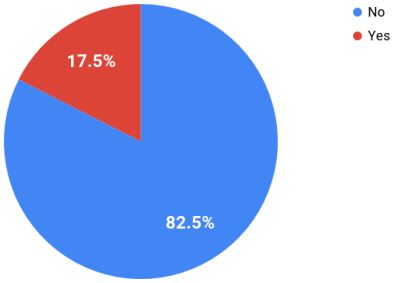
In the future, we will explore the synthesis of feedback on other platforms such as Facebook and Instagram. Further more we will investigate how we can use similar techniques in other areas such as Marketing and Gaming.

6.3 Acknowledgment

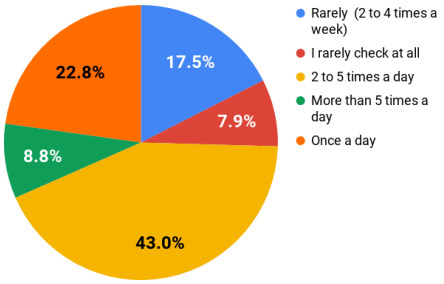
This work is based on research supported by the National Research Foundation (NRF) of South Africa (Grant Number: 119041). Any Opinion, findings and conclusions or recommendations expressed in this material are those of authors and therefor the NRF does not accept liability in regard thereto. Special thanks to Bridge Labs Inc, Johannesburg, South Africa for supporting this work with a Travel Grant. The Second Author acknowledges his Research Assistant, Nikita Patel, for her contribution in drawing the rich pictures in this paper.



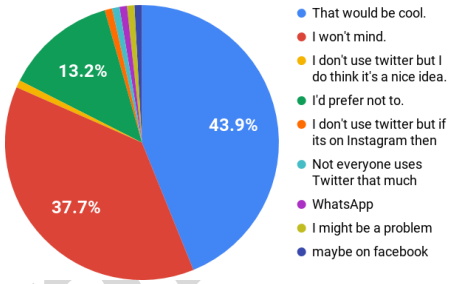
(a) Age Group



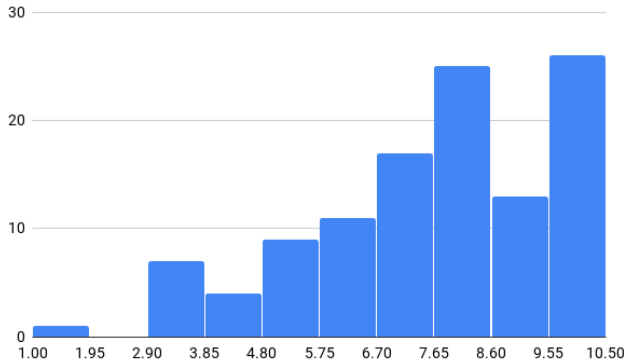
(b) Do instructors us Social Media to support teaching



(c) Frequency of accessing their student email



(d) Prefer to get class announcements on Social Media



(e) On a scale of 1 to 10 how frequently do students use Social Media

Fig. 5. Survey: Opinion about Receiving Educational Feedback on Social Media

References

1. Andreas M Kaplan and Michael Haenlein. Users of the world, unite! the challenges and opportunities of Social Media. *Business horizons*, 53(1):59–68, 2010.

2. Homero Gil de Zúñiga, Nakwon Jung, and Sebastián Valenzuela. Social Media use for news and individuals' social capital, civic engagement and political participation. *Journal of computer-mediated communication*, 17(3):319–336, 2012.
3. W Glynn Mangold and David J Faulds. Social Media: The new hybrid element of the promotion mix. *Business horizons*, 52(4):357–365, 2009.
4. Andreas M Kaplan and Michael Haenlein. Social Media: back to the roots and back to the future. *Journal of Systems and Information Technology*, 14(2):101–104, 2012.
5. Huseyin Bicen and Nadire Cavus. Twitter usage habits of undergraduate students. *Procedia-Social and Behavioural Sciences*, 46:335–339, 2012.
6. Abu Elnasr E Sobaih, Mohamed A Moustafa, Parvis Ghandforoush, and Mahmood Khan. To use or not to use? Social Media in Higher Education in Developing Countries. *Computers in Human Behaviour*, 58:296–305, 2016.
7. Paige Abe and Nickolas A Jordan. Integrating Social Media into the Classroom Curriculum. *About Campus*, 18(1):16–20, 2013.
8. Winner Dominic Chawinga. Taking Social Media to a University Classroom: teaching and learning using twitter and blogs. *International Journal of Educational Technology in Higher Education*, 14(1):3, 2017.
9. Stuart Hepplestone, Graham Holden, Brian Irwin, Helen J Parkin, and Louise Thorpe. Using technology to encourage student engagement with feedback: a literature review. *Research in Learning Technology*, 19(2), 2011.
10. Eva Kassens-Noor. Twitter as a Teaching Practice to Enhance Active and Informal Learning in Higher Education: The case of Sustainable Tweets. *Active Learning in Higher Education*, 13(1):9–21, 2012.
11. Thomas Menkhoff, Yue Wah Chay, Magnus Lars Bengtsson, C Jason Woodard, and Benjamin Gan. Incorporating Microblogging Tweeting in higher Education: Lessons learnt in a knowledge management course. *Computers in Human Behaviour*, 51:1295–1302, 2015.
12. D Gachago and E Ivala. Social Media for enhancing student engagement: the use of Facebook and blogs at a University of Technology. *South African Journal of Higher Education*, 26(1):152–167, 2012.
13. Irshad Hussain. A Study to Evaluate the Social Media Trends among University Students. *Procedia-Social and Behavioral Sciences*, 64:639–645, 2012.
14. Ying Tang and Khe Foon Hew. Using Twitter for Education: Beneficial or simply a waste of time? *Computers & Education*, 106:97–118, 2017.
15. Margaret D Roblyer, Michelle McDaniel, Marsena Webb, James Herman, and James Vince Witty. Findings on Facebook in higher education: A comparison of college faculty and student uses and perceptions of social networking sites. *The Internet and higher education*, 13(3):134–140, 2010.
16. Anabela Gomes and António José Mendes. Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE*, volume 7, 2007.
17. Essi Lahtinen, Kirsti Ala-Mutka, and Hannu-Matti Järvinen. A study of the difficulties of Novice Programmers. *ACM SIGCSE Bulletin*, 37(3):14–18, 2005.
18. V Renumol, S Jayaprakash, and D Janakiram. Classification of cognitive difficulties of students to learn computer programming. *Indian Institute of Technology, India*, page 12, 2009.
19. Susan Bergin, Aidan Mooney, John Ghent, and Keith Quille. Using Machine Learning techniques to predict introductory programming performance. *International Journal of Computer Science and Software Engineering (IJCSSE)*, 4(12):323–328, 2015.

20. Sohail Iqbal Malik. Role of adri model in teaching and assessing novice programmers. Technical report, Deakin University, 2016.
21. Phit-Huan Tan, Choo-Yee Ting, and Siew-Woei Ling. Learning difficulties in programming courses: undergraduates' perspective and perception. In *2009 International Conference on Computer Technology and Development*, volume 1, pages 42–46. IEEE, 2009.
22. Anthony Robins, Janet Rountree, and Nathan Rountree. Learning and Teaching Programming: A review and discussion. *Computer science education*, 13(2):137–172, 2003.
23. Iain Milne and Glenn Rowe. Difficulties in learning and teaching programming: views of students and tutors. *Education and Information technologies*, 7(1):55–66, 2002.
24. Sohail Iqbal Malik and Jo Coldwell-Neilson. A model for Teaching an Introductory Programming course using adri. *Education and Information Technologies*, 22(3):1089–1120, 2017.
25. Cécile L Paris, William R Swartout, and William C Mann. *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, volume 119. Springer Science & Business Media, 2013.
26. Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. Cambridge university press, 2000.
27. Dhiraj Murthy. *Twitter*. Polity Press, 2018.
28. Tamara A Small. What the hashtag? a content analysis of canadian politics on twitter. *Information, communication & society*, 14(6):872–895, 2011.
29. Ricardo Buettner. The Utilization of Twitter in Lectures. In *GI-Jahrestagung*, pages 244–254, 2013.
30. Twitter. Post, retrieve and engage with tweets. `https://developer.twitter.com/en/docs/tweets/post-and-engage/overview`. Accessed: 2019-02-05.
31. Abejide Ade-Ibijola. Synthesis of Social Media Profiles using a Probabilistic Context-Free Grammar. In *2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech)*, pages 104–109. IEEE, 2017.
32. Kaori Kumagai, Ichiro Kobayashi, Daichi Mochihashi, Hideki Asoh, Tomoaki Nakamura, and Takayuki Nagai. Human-like Natural Language Generation Using Monte Carlo Tree Search. In *Proceedings of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation*, pages 11–18, 2016.
33. Daniel Bauer and Alexander Koller. Sentence Generation as planning with Probabilistic ltag. In *Proceedings of the 10th International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 10)*, pages 127–134, 2010.
34. Laurence Danlos, Aleksandre Maskharashvili, and Sylvain Pogodalla. Interfacing Sentential and Discourse tag-based Grammars. In *Proceedings of the 12th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+ 12)*, pages 27–37, 2016.
35. G. Obaido, A. Ade-Ibijola, and H. Vadapalli. Generating Narrations of Nested SQL Queries Using Context-Free Grammars. In *2019 Conference on Information Communications Technology and Society (ICTAS)*, pages 13:1–6. IEEE, March 2019.
36. James Ryan, Ethan Seither, Michael Mateas, and Noah Wardrip-Fruin. Expressionist: An authoring tool for In-game Text Generation. In *International Conference on Interactive Digital Storytelling*, pages 221–233. Springer, 2016.
37. Abejide Ade-Ibijola. Syntactic Generation of Practice Novice Programs in Python. In *Annual Conference of the Southern African Computer Lecturers' Association*, pages 158–172. Springer, 2019.

38. Abejide Ade-Ibijola. Synthesis of Regular Expression Problems and Solutions. *International Journal of Computers and Applications*, pages 1–17, 2018.
39. Alfred V Aho, Ravi Sethi, and Jeffrey D Ullman. Compilers, Principles, Techniques. *Addison wesley*, 7(8):9, 1986.