The Teacher as a Facilitator for Learning
Flipped Classroom in a Master's Course on Artificial Intelligence

Robin T. Bye

Software and Intelligent Control Engineering Laboratory, Department of ICT and Natural Sciences, Faculty of Information Technology and Electrical Engineering, Norwegian University of Science and Technology NTNU, Postboks 1517, NO-6025 Ålesund, Norway

Keywords: Flipped Classroom, E-Learning, Active Learning, Constructive Alignment, Problem-Solving, CodinGame, edX, C-4 Dynamite for Learning.

Abstract: In this paper, I present a flipped classroom approach for teaching a master's course on artificial intelligence. Traditional lectures in the classroom are outsourced to an open online course to free up valuable time for active, in-class learning activities. In addition, students design and implement intelligent algorithms for solving a variety of relevant problems cherrypicked from online game-like code development platforms. Learning activities are carefully chosen to align with intended learning outcomes, course curriculum, and assessment to allow for learning to be constructed by the students themselves under guidance by the teacher, much in accord with the theory of constructive alignment. Thus, the teacher acts as a facilitator for learning, much similar to that of a personal trainer or a coach. I present an overview of relevant literature, the course content and teaching methods, and a recent course evaluation, before I discuss some limiting frame factors and challenges with the approach and point to future work.

1 INTRODUCTION

The flipped, or inverted, classroom teaching methodology can perhaps most simply be defined as swapping learning activities that traditionally have taken place in-class with learning activities that traditionally have taken place out-of-class (Lage et al., 2000). However, as noted in a survey of research on flipped classroom by Bishop and Verleger (2013), flipped classroom constitutes much more than a mere reordering of activities performed at home or in class. They give a more useful definition as the flipped classroom being "an educational technique that consists of two parts: interactive group learning activities inside the classroom, and direct computer-based individual instruction outside the classroom." Elaborating on this definition, Bishop and Verleger (2013) highlights human interaction as the key component of the in-class activities, with a foundation in student-centred learning theories such as those of Piaget (1968) and Vygotsky (1978), whereas explicit instruction methods based on teacher-centred learning theories are outsourced to automated computer technology.

Similarly, Abeysekera and Dawson (2015) emphasise that in a flipped classroom, "the information-transmission component of a traditional face-to-face lecture is moved out of class time [...and replaced with] active, collaborative tasks." Consequently, students prepare for class by outside-class activities normally covered by traditional lectures, thus freeing up valuable in-class time to student-centred learning activities. After class, students can continue working on in-class tasks they did not finish, explore some topics in more detail, revise material, and further consolidate knowledge (Abeysekera and Dawson, 2015).

1.1 Active Learning

According to Sotto (2007), higher education is dominated by the *transmission method* of teaching, which can be popularly rephrased as teaching by telling. Synthesising research on the effectiveness of traditional lectures, Bligh (1998) shows that they are not very effective for developing skills, values or personal development, all of which are natural learning goals in higher education. Flipped classroom moves this one-way passive learning activity outside the classroom and replaces it with active learning, which can be defined as any teaching method that engages the students in the learning process (Prince, 2004). However, pointed out by Bishop and Verleger (2013), this definition could in principle also include lectures

Bye, R.
The Teacher as a Facilitator for Learning - Flipped Classroom in a Master's Course on Artificial Intelligence.
DOI: 10.5220/0006378601840195
In *Proceedings of the 9th International Conference on Computer Supported Education (CSEDU 2017) - Volume 1*, pages 184-195
ISBN: 978-989-758-239-4

where students reflect, take notes, and ask questions. Thus, to maintain a contrast with teacher-centred learning, it can be useful to separate lecture-based methods from traditional lectures and define active learning as students participating in the learning process, doing more than just passive listening (Bonwell and Eison, 1991). In line with this, active learning is indeed an inseparable part of flipped classroom (Schaathun and Schaathun, 2016).

Bishop and Verleger (2013) identify several active learning paradigms, such as *constructivism* and *collaborative learning* (originating from the theory of cognitive conflict by Piaget (1968)), *cooperative learning* (based on the theory of Vygotsky (1978) on the zone of proximal development), *peer-assisted learning* (defined by Topping and Ehly (1998) as "the acquisition of knowledge and skill through active helping and supporting among status equals or matched companions"), and peer-tutoring (e.g., see Topping, 1996, for a review). Another active learning method is *problem-based learning*, which has overlap with learning methods under the umbrella of peer-assisted learning but can also be done individually (Bishop and Verleger, 2013).

Several metastudies show that active learning in science, technology, engineering, and mathematics (STEM) has several advantages regarding performance, ability to reproduce material, and motivation and engagement (e.g., Prince, 2004; Schroeder et al., 2007; Freeman et al., 2014), whilst cooperative learning strategies have been shown to be particularly effective for achieving deep learning (e.g., Foldnes, 2016; Bowen, 2000; Johnson et al., 1998; Springer et al., 1999).

1.2 Constructive Alignment

For the last decades, there has been a dramatic change in higher education worldwide, with many more students enrolling, from a wider diversity of background, and with a broader range of approaches to learning (Biggs and Tang, 2011). In engineering, students of today conceive learning as anything between simple memorisation of definitions, applying equations and procedures, or making sense of physical concepts and procedures, to seeing phenomena in the world in a new way or changing as a person (Marshall et al., 1999). The big variation among students in this taxonomy ranging from *surface learning* (memorisation) to *deep learning* (learning associated with understanding and ultimately changing as a person), with an added category of *strategic learning*, in which students aim for good grades with minimal effort (Entwistle and Ramsden, 1983), has necessarily had an impact on how higher education is being taught (Felder and Brent, 2005).

Because it is well documented that students' approaches to learning has a significant effect on achieving learning outcomes (e.g., Gynnild, 2001; Marton, 1981), many studies have tried to identify factors that promote deep learning (e.g., Marton and Booth, 1997; Prosser and Trigwell, 1999). A popular answer for dealing with the challenges of today's diverse student mass is the theory of *constructive alignment* (Biggs, 1996). Constructive alignment (CA) merges the constructivist view that students learn by doing with aligning the teacher, the students, the teaching context, the learning activities, and the learning outcomes (Biggs and Tang, 2011). This is achieved by working backwards when designing a course, starting with the *intended learning outcomes* (ILOs), defining assessment tasks closely related to the ILOs, and finally choosing teaching methods and learning activities aligned with the ILOs and assessment tasks (Biggs and Tang, 2011).

Biggs (e.g., Biggs, 1996, 2011; Biggs and Tang, 2011) is especially concerned about the assessment tasks. Since many students are strategic in their approach to learning, the exam (assessment tasks) effectively defines the curriculum, as only those parts of the course that will affect their final grade will be prioritised. The answer in CA is to incorporate assessment tasks with grades well aligned with ILOs to ensure students achieve the latter.

1.3 Cognitive Load Theory

As noted in the literature review by Schaathun and Schaathun (2016), active learning adopts the constructivist view that learners must construct their own knowledge. However, at the same time, in order to move new information from a working memory with very limited capacity into long-term memory, deep cognitive processing is required (Anderson, 2015). This process, referred to as *cognitive load theory* (Clark et al., 2011), represents a bottleneck in learning when to complex problems and information must be processed in too short a time. A remedy suggested by several authors (e.g., see Sotto, 2007) is to break problems into small, manageable exercises in order to reduce the cognitive load. In flipped classroom, where typically online video lectures are offered outside the classroom, students can manage this cognitive load by regulating pace and rewatching video lectures (Abeysekera and Dawson, 2015), as well as completing their other learning tasks in their own order of preference, with possible repetitions if necessary.

1.4 Facilitating Learning

According to Biggs (2011), the teacher must create a learning environment that facilitates learning activities that in turn make the students achieve the desired learning outcomes. In CA, the key to success is to make sure that all teaching and learning components such as the curriculum and the ILOs, the teaching methods, and the assessment tasks are aligned to each other. However, it is commonly accepted that lack of self-monitoring and self-regulation among students will lead to poor academic results (e.g., Lan, 1996; Borkowski and Thorpe, 1994). Consequently, the learning environment itself is not sufficient to achieve ILOs, since students' individual skill in self-monitoring and self-regulation, that is, selecting and structuring the material to be learnt, will highly affect the learning outcomes (Gynnild et al., 2008). To cope with this challenge, (Gynnild et al., 2007) suggests that the teacher must adopt a role as a *facilitator for learning*, much similar to a personal trainer at the gym, guiding the trainee to do the right exercises, adjusting the "weights," or cognitive load, whilst encouraging and supporting the trainee, and eventually making the trainee self-monitored and self-regulated.

Obviously, this is a bold ambition, and increasingly so with large classes, however, flipped classroom may just be the tool that makes it possible.

1.5 Outline

In the following sections, I first give an overview of a master's course on artificial intelligence (AI) in which I have employed flipped classroom. Next, I present the teaching methods I have used and present a course evaluation that includes feedback from students in the 2016 cohort. Finally, I discuss some limiting frame factors and challenges to my approach, point to future work, and draw some conclusions.

2 COURSE OVERVIEW

The elective third-semester master's course *IE502014 Topics in Artificial Intelligence (TAI)* has been run with two different cohorts (autumn 2015 and 2016) since the inauguration in autumn 2014 of a new master of science programme in simulation and visualization at NTNU in Ålesund. The course consists of 7.5 ECTS[1] credits and runs for 14 weeks, where all in-class activities are constrained to a single weekly teaching day, referred to as workshops. Providing an introduction to the field of AI and a number of selected topics therein relevant for solving real-world problems, students study AI with respect to

- *modelling* a variety of problems in suitable state space
- *design* and *implementation* of intelligent search, optimization, and classification algorithms
- *simulation* and *testing* of models and algorithms
- *visualisation* and *analysis* of the results

The ILOs of TAI are defined within the three categories of knowledge, skills, and general competence. Specifically, upon completion of the course, students should be able to

- describe AI as the analysis and design of intelligent agents or systems
- explain terms such as perception, planning, learning, and action as fundamental concepts in AI
- define problems in suitable state space depending on choice of solution method
- solve problems by means of search methods, evolutionary algorithms, swarm algorithms, neural networks, or other AI methods
- collect information from scientific publications and textbooks and reformulate the problem, choice of methods, and results in a short, concise manner
- discuss and communicate advantages and limitations of AI as a science

The course is split in two parts provided by two teachers. Part A, taught by me, focusses on solving problems by means of intelligent agents and the use of various search algorithms, dealing with both uninformed and informed (heuristic) search, local search, and adversarial search, whereas Part B, taught by my colleague Associate Professor Ibrahim A. Hameed, focusses on optimisation and constraint satisfaction problems, as well as machine learning topics such as fuzzy expert systems, artificial neural networks (ANN), and hybrid intelligent systems. In relation to this paper, the teaching methodology I apply for Part A is the most relevant but I may refer to Part B or the course as a whole where appropriate.

TAI has the following main components:

- course textbooks and scientific literature
- the learning management system (LMS) Fronter
- three mandatory assignments
- final oral exam
- online code development platforms
- massively open online courses (MOOCs)
- full-day in-class workshops

In the following, I will present each of these components in turn.

[1] European Credit Transfer and Accumulation System.

2.1 Textbooks and Topics

The course relies heavily on the following textbooks:

- *Artificial Intelligence: A Modern Approach (AIMA)* (Russell and Norvig, 2013),
- *Artificial Intelligence: A Guide to Intelligent Systems (AIGIS)* (Negnevitsky, 2005), and
- *Learning for Data: A Short Course (LFD)* (Abu-Mostafa et al., 2012).

Both times we have run the course, the following ten topics in the textbooks were studied (topics 1–5 constitute Part A, topics 6–10 constitute Part B):

1. Introduction to AI (AIMA Ch. 1)
2. Intelligent Agents (AIMA Ch. 2)
3. Solving Problems by Searching (AIMA Ch. 3)
4. Beyond Classical Search (AIMA Ch. 4)
5. Adversarial Search (AIMA Ch. 5)
6. Constraint Satisfaction Problems (AIMA Ch. 6)
7. Fuzzy Expert Systems (AIGIS Ch. 2–4, case study Ch. 9)
8. Artificial Neural Networks for Pattern Recognition (AIGIS, Ch. 6, case studies Ch. 9)
9. Hybrid Intelligent Systems (AIGIS Ch. 8, case studies Ch. 9)
10. Nonlinear Transformation and Feature Extraction (LFD Ch. 3)

2.2 Fronter

Fronter[2] is the official LMS in use at NTNU in Ålesund. Every course has its own "room" in Fronter that enrolled students can access. The course room for TAI contains a Frontpage with a summary of all relevant information that is continously being posted, including a welcome message; information about where to buy the course textbooks; and course material (course overview, oral exam information, assignment handouts, and workshop material). The Frontpage also contains internal hyperlinks to the course material (usually pdfs) contained in a well-structured Documents directory. Students can quickly and easily access the material from the Frontpage, which is important if using a smartphone, but can also choose to navigate the Documents directory for complete access to all material. Assignments are handed in through the Fronter submission system and detailed feedback on students' performance is also posted there. Using the Fronter news functionality, we inform students whenever new material is available or when already posted material has been updated.

The teachers have tried to encourage the Fronter forum functionality by posting answers to several questions that students ask in class or by email but there has been zero activity from students in the forum. This is likely due to students being on-campus and seeing each other steadily, thus reducing the need for an online forum. Still, the teachers find the forum functionality useful but then acting more as a knowledge bank of frequently asked questions (FAQs) with answers.

2.3 Assignments and Exam

Three mandatory assignments (two for Part A, one for Part B) must be passed for permission to enter the end-of-semester final oral exam. The assignments typically consist of both theoretical short answer questions and project-like programming exercises. Employing a grading scheme consisting of the letters A to F, where A is best and F is fail, a pass grade is awarded if the work has the quality of a C or better. However, the grade on assignments does not contribute to the final grade, which is determined solely by a final individual oral exam. Students usually get 3–4 weeks to complete and submit their assignments on Fronter.

Individual assignment reports, as well as a laptop with computer code, can be brought to the exam to provide individual entry points of discussion. The oral exam is 25 mins and students are examined by the course teachers.

2.4 Code Development Platforms

Developing a framework with simulation environments in which to test intelligent algorithms can be a time-consuming task and divert attention from the learning outcomes that we want the students to achieve. For the programming exercises for Part A, I rely on two online websites that provide such frameworks, namely HackerRank[3] and CodinGame.[4] Students have to register a user (free of charge) on both websites.

Using HackerRank and CodinGame removes the need for writing a lot of boilerplate code and simulation logic, leaving the students to focus on the design and implementation of the algorithms. Both websites support more than 20 of the most common programming languages, provide discussion forums, blogs, and leaderboards (rankings), with CodinGame also providing entertaining gamelike visualisations of the code execution, which is quite useful when testing an algorithm. Thus both websites provide fun and playful environments that act as a catalyst for learning

[2] http://www.itslearning.eu/fronter

[3] https://www.hackerrank.com

[4] https://www.codingame.com

where students can compete both against themselves and others in the quest of obtaining better scores.

2.4.1 HackerRank

HackerRank provides numerous programming challenges in several domains and programming languages, including the domain most relevant for this course, namely AI. All challenges consist of a problem description and a code window in a web browser where students can enter their programmes, or they can use a plugin to enable writing their code locally in their own editor. Programmes are then tested on text input and must produce the correct text output. Sample code to get students started is provided in many cases.

For Part A of TAI, I cherrypick exercises from relevant AI subdomains on HackerRank, including Graph Theory, A* Search, Bot Building, Alpha Beta Pruning, Combinatorial Search, and Games. An example exercise is BotClean Large, in which students must program an *intelligent agent* (bot) to clean all the dirty cells in a 2D grid. After submitting their code, a simulation tests the bot of each student on a number of different scenarios and returns a score.

2.4.2 CodinGame

CodinGame works in a similar manner as HackerRank but also provides game-like 2D graphical feedback of the behaviour of the code. The stub code to get students started on a particular programming challenge is usually better than the ones on HackerRank. Programming challenges are divided into single-player challenges (ranked Easy, Medium, or Hard) that can be found in the Training category, and larger, multiplayer games called Bot Programming, that can be found in the Multiplayer category (past contents) or Contests (currently running contests). One example of a multiplayer game is Game of Drones, in which students must control a fleet of drones and compete against other players (classmates, online players, or the default AI provided by CodinGame) with their own fleet of drones in a battle of controlling as many zones as possible in a large 2D grid.

2.5 Video Lectures and E-learning

With the advent of MOOCs, numerous online e-learning resources exist, often free of charge. Two of the leaders in the field of MOOCs are Udacity[5] and edX.[6]

Udacity was perhaps the very first MOOC and was born out of a Stanford University experiment by famous AI researchers Sebastian Thrun and Peter Norvig (incidentally, Norvig is also co-author of the AIMA course textbook (Russell and Norvig, 2010)). They offered an online course called *Introduction to Artificial Intelligence* online for free and managed to achieve a simultaneous enrolment of more than 160,000 students in more than 190 countries.

EdX was founded by the Massachusetts Institute of Technology (MIT) and Harvard University in May 2012 and provides courses from more than 60 universities, corporations and institutions.

We encourage students to enroll for free in one or more of the courses provided by Udacity, and *demand* that they enroll, also for free, in the edX course *CS188.1x Artificial Intelligence* prepared by staff at the University of California Berkeley. The courses that we suggest build heavily on the textbook by Russell and Norvig (2010) and other textbooks on AI and contain numerous interactive video lessons, self-tests such as quizzes and multiple-choice tests, programming exercises, and other material that encourage active learning and are useful for TAI.

2.6 Workshops

All in-class learning activities in TAI take place in an ordinary flat classroom on a single weekday that typically begins at 8:15 and ends at 16:00. However, the teacher is neither required nor expected to be present during the entire day.

The purpose of calling this teaching day a *workshop* is to emphasise an active, collaborative, student-centred learning environment in which students will construct their own learning, in contrast with more teacher-centred learning environments. I will discuss more about learning activities in the following section.

3 TEACHING METHODS

3.1 Information and Structure

There are three important bits of information that must be distributed to students before the very first workshop.

First, because the course is closely aligned with the course textbooks, it is vital to let the students know which books to buy as long as possible before

[5]http://www.udacity.com

[6]http://www.edx.org

the start of the semester. Unfortunately, some students enrol in courses quite late, and we therefore have to re-send this information several times to make sure latecomers get it.

Second, we post a detailed 10-page course overview, which serves as a reference guide for the entire semester. This document includes core information, high-level ILOs, teaching methodology, reading material and topics, a weekly schedule, and information about assignments, online resources, and the course evaluation.

Third, I post the description of my first workshop, with details on prerequisite homework (getting textbooks and registering on online resources) as well as in-class learning activities. Unsurprisingly, many students fail to do the homework and the workload of the workshop is therefore intentionally smaller than the remaining workshops to allow for this, with no homework reading, exercises, or video lectures.

Early in the semester we also post a detailed list of potential oral exam questions. As we elaborate on in an accompanying paper submitted to this very conference (Osen and Bye, 2017), posting such a list serves several purposes. It reduces uncertainty and stress related to the exam but just as important, it provides students with mental hooks that aid in constructing knowledge as they progress through the course, since having read the questions will reduce the cognitive load when new material is introduced, and they will be able to more quickly putting new knowledge into context and storing it in long-term memory.

All of the above information is posted on Fronter in a highly structured way for easy and quick access, even when using smartphones and tablets. The course overview is rich in detail but is a document that we encourage students to refer back to regularly. It is useful for the students to have a schedule of when all the different topics will be introduced as well as assignment deadlines as early as possible so that students can plan their studies and reserve time for assignments. Introducing the teaching methodology with the emphasis on flipped classroom and active learning, with overarching ILOs and a well-defined main objective of the course, also provides a useful top-down view that guides the students during the course.

3.2 Workshops

Some time between a workshop and the next (typically 3–5 days), a new workshop description is released on Fronter. Each workshop description contains details such as the date, classroom, start time, teacher, and textbook chapters/topics to be studied; the ILOs of that particular workshop; a rough schedule of tasks to be completed; details on each task; and homework for the next workshop. This homework consists of reading selected chapters and solving selected exercises in the textbook, as well as completing various tasks in the online edX course (see Section 3.3) or on the code development platforms (see Section 3.4).

Every workshop begins with a short evaluation of the current status regarding problems students have faced in their homework, questions they may have, what they think about the teaching methods, and other things that come to mind. Typically, there will be some parts from the homework that students have struggled with and have questions about. Minor questions are answered immediately, however, if it is clear that a more thorough explanation is needed, the teacher writes down the questions or topics on a list and typically gives a *micro-lecture* on the various topics afterwards.

Next, students are given a number of active learning tasks, varying in size. For example, a two-minute task could be to discuss with another person in class the possible answers of a question I give them, whilst students can be working for up to several hours on an larger exercises or assignments.

The teacher is not required to be available at all times. Indeed, it may be beneficial to leave the classroom at times, as internal discussion and group work seem to flourish with the teacher absent.

3.2.1 Self-tests

For *self-tests* (quizzes or multiple-choice questions), I usually ask everyone to do them individually, but encourage discussions with other students and myself. This has the advantage that those who prefer some time on their own to reflect on the answer get that, whereas those who prefer to interact with others in search of answers can do so. Afterwards, I ask students in turn for answers, and most importantly, the reasoning behind the answers. If the student's reasoning is weak, I ask for clarifications from the rest class, and finally, I often provide my own explanation, in different wording.

Digital self-tests has the advantage that *immediate feedback* is returned to the students. However, they also require questions to be unambiguously phrased. During a semester, there will very often be students who are dissatisfied with a question, claiming that another answer is more correct, or that there is no right answer. This should not be viewed as something unfortunate. Rather, it is a golden opportunity for discussion in class and a driver for deep learning. Students must have a thorough understanding of the topic of the question and be able to formulate to the teacher why they disagree with the question, and other students have to opportunity to delve into the discussion.

3.2.2 Oral Presentations

For *oral presentations*, students will generally be working in groups, which tend to lower the fear of presenting and generally improves the quality of both quiz answers and presentations. Each group will randomly be assigned a topic for the presentation, for example, a particular search algorithm. Knowing that they will have to make a presentation in class but not knowing the topic beforehand likely acts as a catalyst for completing the homework in a serious manner. Also, making the presentation ad hoc in class and explaining the topic to others force the students to organise their knowledge and insight on the topics.

Naturally, the students' presentations will suffer from the short preparation time in the classroom. I therefore interrupt the presentations when needed and first ask the presenting students if they are able to clarify the issue at hand, and if not, I ask the class as a whole if someone can help out. Even if I get a good answer, I usually rephrase the answer with my own words and try to be as concise and precise as possible, and hopefully, both the presenters and the rest of the class get a much clearer picture of the particular issue.

3.2.3 Textbook Exercises

Another activity during workshops include discussing answers to *textbook exercises*. These exercises differ from typical online exercises and self-tests, as they often require more work and are more time-consuming; one often have to make an analysis of the problem first, and then provide a long and elaborate answer. It would therefore be a waste of time to do such exercises in class, and instead, students are given a short list of recommended textbook exercises to complete at home before each workshop. In class, answers to the exercises are walked through and discussed as needed.

3.2.4 Coding Challenges

Finally, being a course on topics in AI, writing computer code and implementing intelligent algorithms is an inherent and important part of the course. During workshops, on average, a large portion of the time is devoted to larger *coding challenges* contained in the assignments, or minor coding challenges that are solvable in a short period of time in class. More details on coding are provided in Section 3.4.

3.2.5 Competitions

To add some spice to the learning environment, I sometimes arrange for small informal *competitions* in the class, where the winner, or winning team, gets nothing more than pride and glory. Sometimes I split the class in groups and let each group work on the same problem for 10-15 mins, say. The first team who submits a working solution gets a point. Points are then accumulated for various learning tasks during the day. The students tend to like this concept but it should not be overdone and detract from the learning process.

3.2.6 Discussions

Discussions are not a scheduled task in the workshop but rather something that happens ad hoc during the tasks mentioned above. Forcing students to discuss a topic is rarely very awarding. Instead, having outsourced the traditional lectures from the classroom provides more time to allow discussions to emerge as needed. This is extremely useful for deep learning, as it helps students learning by reducing the cognitive load, delving deeper into topics, and allowing time to organise and store new material in long-term memory. In addition, discussions will often sidetrack into related topics, evident of students looking at the world from a different perspective, discovering relationships across topics, and even changing as a person.

3.3 Online AI Course

All students in TAI are required to enrol in an online AI course offered by edX called *CS188.1x Artificial Intelligence*. This course has been archived since 2014, which means that no teaching staff maintain or are active in the course. Nevertheless, all the resources provided are available, free of charge, to edX users. In Part A of TAI, four of the five topics are covered by the edX course, with excellent video lectures, step-by-step tutorials, and quizzes. The video lectures come both in long and unedited versions and in short and condensed edited versions that are also interspersed with self-tests with immediate feedback, all easily accessible on portable devices such as mobiles and tablets. The voices in the videos are transcribed to text, which makes it easy for students to quickly browse through a video by scrolling through the text until a given point of interest.

Whereas some students favour watching the long videos in one go and then use the the short, edited videos for repetition, others jump straight to the edited videos, especially if they have studied the textbook first. The video format and interactive material also make it easy for students to squeeze in short micro-sessions of homework, e.g., during a 15-min bus ride.

3.4 Coding Challenges

To construct knowledge, skills, and competence within the AI topics that we study, students need to practice on modelling problems with relevance to the real-world and be able to select or modify existing intelligent algorithms from the AI literature or devise new algorithms suitable to the level of abstraction in their models. The next step is to implement the algorithms and test them in a simulated environment, and perform an analysis of their behaviour, often requiring visualisations to understand what is going on.

Doing this from scratch requires both skills and a lot of time, and henceforth we have outsourced this to two code development platforms, HackerRank and CodinGame, both of which provide excellent frameworks within the students do not have to worry all the necessary boilerplate code and instead focus on the algorithms.

3.4.1 Intelligent Agents

Both platforms have been designed in a similar manner where the user must input a computer programme, commonly referred to as the AI, or *bot*, that reads data from standard input, does some processing, and then outputs a desired action to standard output. This loop repeats for a number of time steps until the simulation, or *game*, ends. Thus, the internals of the simulated environment behaves much like a black box to the bot, however, certain parameters in the environment are "sensed" and provided to the bot at each time step, and in addition, each coding challenge provides details of the model the simulated environment is based upon sufficient that a useful bot can be constructed. Hence, the coding challenges adopts the same paradigm as the course textbook, in which AI is centred around *intelligent agents* that sense, plan, and act (Russell and Norvig, 2010).

Coding challenges typically revolve around certain themes in computer science and AI such as queues, lists, search algorithms, path planning, control, etc. Very often, each challenge comes in several versions, Easy, Medium, and Hard, in which the challenge becomes gradually more complex and difficult to solve. An example is the Mars Lander challenge on CodinGame, where students doing the Easy level must design an intelligent agent, first for controlling the landing of a spacecraft in the vertical dimension only, then, for the Medium and Hard levels, in two dimensions, which is exceedingly harder.

3.4.2 Design Process

When working on coding challenges, I emphasise the importance of thinking about the problem before jumping straight into coding. Together with the students, we brainstorm various potential models of the problem to be solved, and discuss aspects such as levels of abstraction, and their advantages and limitations. The solution method to be used is dependent on the model, as an intelligent algorithm requires the problem to be represented by exactly those components that the algorithm was designed for. The students then run a number of tests on the coding platform to verify their design and receive automatic feedback on which tests were passed or failed.

In contrast with HackerRank, the CodinGame platform also provides *visualisations* of the simulated environment. This is very valuable to students in the iterative process of testing, debugging, refining, and analysing their algorithms, because visual feedback quickly can reveal problems in the execution of the algorithm.[7]

4 COURSE EVALUATION

To conform with the requirement in NTNU's Quality System for Education that a *course evaluation* must take place every time a course is run, we adopt a scheme where all students present during workshops (typically 7–8 students) belong to a so-called reference group. At the beginning of a workshop, we begin with a class discussion about course-specific issues such as status, progression, problems, etc., as well as general issues you about the master programme in general. The teacher takes notes, which form the basis of an end-of-semester course evaluation report. These short meetings provide an important arena and opportunity for students to help the teacher adjust course early before issues become too big to fix.

In the sections below, I highlight some of the evaluation findings from the TAI course that was run autumn 2016.

4.1 Workload

Students thought the overall workload was appropriate, although perhaps slightly bigger than in some other courses in the master programme. Some students thought the workload for the assignments was

[7]For this reason, and other minor reasons, we have chosen to use the CodinGame platform exclusively starting from 2017.

high but thought this was justified by the assignments being highly valuable. For example, one student said that "*(...) the assignments were big but good. Even though this course had a high workload, in retrospect I would say that this was necessary, and the main reason I learnt so much.*"

4.2 Teaching Methods and Learning Environment

Students expressed enthusiasm for the workshop format and the elements of flipped classroom and had little to add apart from comments that when revising theory from homework, e.g., in a micro-lecture, it would be slightly better for the teacher to have slides than having to move back and forth in the online video lecture. Students also appreciated the fact that videos were available in both short and long versions, and different students sometimes favoured one over the other. Students also expressed that the self-tests were appropriate and useful, however, some multiple-choice questions could contain ambiguously phrased questions and therefore one could argue which answer was really the correct one.

Such input is very useful for both the teacher and for fellow students, as it triggers class discussion and deeper insight into the problem that is being studied. This is also an example of deep learning, commonly observed with skilled students, as more shallow surface learners will fail to note such ambiguities in the questions.

Regarding the code development platforms, there was consensus that HackerRank should not be used, mainly because it was more difficult and lacks visualisations, in contrast with CodinGame. Students also thoroughly enjoyed the *gamification* part of CodinGame, where one achieves rating points and badges and therefore are continuously eager to improve.

4.3 Assignments

The students thought the topics covered in assignments and the design of the problems to be solved were appropriate, although the workload was quite high. They also appreciated that assignments are first released as draft versions and then modified by the teacher and the students together through constructive class discussions.

4.4 Structure and Information

The students expressed satisfaction with the course structure and the continuous flow of information by means of Fronter. The teachers have tried to encourage the Fronter forum functionality by posting answers to several questions that students ask in class or by email but there has been zero activity from students in the forum. Students emphasised that being on-campus and seeing each other regularly face-to-face, there was not much need for an online forum. Still, the teachers think the forum functionality should be used but then acting more as a *knowledge bank* of frequently asked questions.

4.5 Summary of Feedback

Throughout the course and in the weekly evaluation sessions involving the entire class, students had very little negative concern about the course itself, be it teaching methods, workload, or curriculum. When they had concerns, this was mainly related to clashes of assignment deadlines across courses and suggestions to reduce workload or modify content of assignments.

Interestingly, students were very eager to discuss issues of other courses and the master programme as a whole. We see this as a sign that this course provides a safe and relaxing learning environment, since students felt comfortable and very keen to share these details with the teachers.

Finally, almost all students told us that they were highly satisfied with the course, with two students expressing in writing that "*I also wish to express that this has been one of the best courses I have taken in higher education*" and "*[t]hank you for teaching one of the better courses if not the best course I've had in this master[']s program[me].*"

5 DISCUSSION

In this paper I have presented a flipped classroom approach for teaching a master's course on AI. Whilst it should be clear that the course has been a great success, the reader should by no means believe that we have found the holy grail for teaching higher education courses. Indeed, there are several limiting frame factors and challenges that must be overcome for flipped classroom to work as intended. Below I discuss some of these.

5.1 Frame Factors

The *number of students* in a class can be a limiting *frame factor* for achieving ILOs. In the course presented here, run in 2015 and 2016, both cohorts consisted of about 10 students, of which 2–3 quit early or never turned up.[8] If the class had exceeded about 25 students (which is also the number of seats funded by the Ministry of Education), it may have been more difficult to achieve the same fruitful learning environment in the classroom. For larger classes, one would likely have to abandon flat classrooms and use large auditoriums. University management would typically see no problem in this and tell teachers to go ahead and do their traditional one-way lectures as usual, which is not desirable. Optionally, one could split such large cohorts into smaller groups, which would multiply the number of teachers required and associated costs. This begs the question why there is a transition in teaching methodology from classroom-style teaching (one teacher per class of 30 students, say) in lower education to auditorium-style lecture-based teaching in higher education, thus turning students into passive learners. Flipped classroom can still be useful for large classes, however, but may require much more effort in facilitating the in-class learning activities.

Another important frame factor is the *available online resources*. In TAI, we have been lucky to find both an edX course that is closely aligned with the topics in the course textbook for Part A, as well as the CodinGame platform that provides a high quality framework with a vast variety of AI coding challenges suitable for achieving ILOs. Producing these online resources oneself with the same quality, as well as the material contained in textbooks, is an impossible task for the common teacher. Still, with *e-learning* as a buzz word nowadays, university management are very eager for teaching staff to make videos to put online. I think this approach is flawed, as it is extremely time-consuming to produce video lectures oneself of the desired quality, not to mention developing a suitable simulation environment for testing algorithms. Instead, teachers should act as *facilitators* and devote their time to finding such resources and tying them together in a didactically sound manner. For example, the teacher must take care in selecting the right coding challenges and guiding the students during problem-solving (see the next section).

5.2 Problem-based Versus Problem-solving Learning

Problem-based learning is a hot topic of higher education and the interested reader may refer to our accompanying paper for reflections on many of its aspects (Osen and Bye, 2017). Here, I refrain myself to a very important finding by Hattie and Goveia (2013), who upon examining 800 meta-analyses notes that problem-based learning cannot be shown to have a positive effect on achieving ILOs! The reason for this, according to Sotto (2007), is the dinstinction between problem-*based* and problem-*solving* learning. He argues that one should use a teaching approach that employs well-designed case studies and avoid problem-based and student-centred learning, especially for larger problems and when there is no clear guidance towards how to solve them. Otherwise, students will get stuck or spend too much time on finding the necessary pieces to solve the puzzle by searching the Internet or studying textbooks.

Adopting this distinction of Sotto (2007), I believe the focus on *problem-solving learning* is a major key to the success we have experienced in our course. For example, the coding challenges on CodinGame are carefully selected and much attention is given during workshops on how to model the problems and how to select appropriate solution methods.

On the other hand, if problems are too rigidly defined, with perhaps only one solution approach possible, as when instructing students to follow detailed step-by-step instructions, there is a danger that creativity is neglected and students only achieve surface learning (Andersen, 2010). Comparing with bachelor's courses, where such rigid schemes to some extent can even be desirable, master's courses should have a higher emphasis on being research-based and investigative in nature.

In TAI, we seem to have found the right *balance* between coding challenges being too rigid versus too open-ended. There are nearly always several paths for solving a given problem, and the teacher often provides a simple outline of a solution as a starting point and then guides students towards improving the solution.

5.3 C-4 Dynamite for Learning

Central to my teaching approach is an emphasis on four axes of learning, **C-4**, that together are dynamite[9] for learning: *creativity*, *cooperation*, *competetition*, and *challenge*.

First, the learning activities we offer in the course should facilitate *creativity*. In a creative process, students model and design solutions to a wide variety of problems but are usually not restricted (much) on how to model the problem and which method to use.

Second, we employ many *cooperative learning* activities in class and we encourage students to cooperate with each other also for individual work.

[8] 15 students are enrolled in 2017.

[9] C-4 is also a common plastic explosive.

Third, as mentioned previously in Section 3.2.5, I sometimes run small, informal *competitions* in class as a driver for motivation. In addition, the CodinGame platform is inherently competitive, as users get rating points as they improve their solutions or solve more problems. Students therefore compete not only against others but also against themselves in a quest for better ratings.

Fourth, we strive for students to critically *challenge* all the concepts they are introduced to in the course and have indeed found that students tend to do this more as the course progresses. Triggers for the students can be as minor as mistakes in the written formulations in quizzes, as discussed in Section 3.2.1, or discovering that a solution method has certain limitations that they first found out about when implementing it on CodinGame. For example, a given intelligent algorithm may not be able to provide a good answer in reasonable time and a particular CodinGame test therefore fails.

5.4 Future Work

In future versions of TAI, my colleague and I will look both at the curriculum and on the learning activities that we offer to bridge the gap between Part A and B of the course. Most of what I have presented in this paper relates to my own flipped classroom approach for teaching of Part A, whereas my colleague has not been able to find the same highly aligned and relevant online resources to be able to facilitate flipped classroom to the same degree for Part B. We will also address the issue of colliding assignment deadlines across parallel courses. Finally, we will be showcasing this course to our colleagues as a motivator for implementing flipped classroom themselves.

5.5 Conclusions

I have shown one way out of many that flipped classroom can be implemented in a master's course on AI. The key to success is to be able to facilitate learning by cherrypicking textbook chapters and relevant literature as well as suitable online resources such as video lectures, self-tests, and coding challenges. When such resources do not exist or lack in quality or theory, one can try to make one's own material, e.g., a compendium supplementary to the textbook, or one's own video lectures. However, this is a very time-consuming process if one wants to achieve the desired quality, and should be a last resort.

I favour a teaching approach that has much in common with constructive alignment, but emphasise that the right balance between too rigidly defined learning activities where students are being "spoonfed" and too vaguely defined problem-based learning activities must be found, so that the problems are investigative in nature with different possible paths towards solutions, yet at least one path must be reasonably easy for the students to find and follow.

ACKNOWLEDGEMENTS

The Software and Intelligent Control (SoftICE) Laboratory is grateful for the financial support given by the Study Committee at NTNU in Ålesund through the project *Research-based and Innovation-driven Learning (FILA)*, grant no. 70440500.

REFERENCES

Abeysekera, L. and Dawson, P. (2015). Motivation and cognitive load in the flipped classroom: definition, rationale and a call for research. *Higher Education Research & Development*, 34(1):1–14.

Abu-Mostafa, Y. S., Magdon-Ismail, M., and Lin, H.-T. (2012). *Learning From Data: A Short Course*. AMLBook.

Andersen, H. L. (2010). "Constructive alignment" og risikoen for en forsimplende universitetspædagogik. *Dansk Universitetspædagogisk Tidsskrift*, 5(9).

Anderson, J. R. (2015). *Cognitive Psychology and Its Implications*. Worth Publishers, 8th edition.

Biggs, J. (1996). Enhancing teaching through constructive alignment. *Higher Education*, 32:347–364.

Biggs, J. (2011). Aligning teaching for constructing learning. The Higher Education Academy, York, United Kingdom. Accessed online on 27.09.2011 at http://www.heacademy.ac.uk/assets/documents/resources/resourcedatabase/id477_aligning_teaching_for_constructing_learning.pdf.

Biggs, J. and Tang, C. (2011). *Teaching for Quality Learning at University*. McGraw Hill/Open University Press, 4th edition.

Bishop, J. L. and Verleger, M. A. (2013). The flipped classroom: A survey of the research. In *ASEE National Conference Proceedings, Atlanta, GA*, volume 30.

Bligh, D. A. (1998). *What's the Use of Lectures?* Intellect Books.

Bonwell, C. C. and Eison, J. A. (1991). *Active Learning: Creating Excitement in the Classroom. 1991 ASHE-ERIC Higher Education Reports.* ERIC.

Borkowski, J. and Thorpe, P. (1994). Self-regulation and motivation: A life-span perspective on under- achievement. In Schunk, D. and Zimmermann, B., editors, *Self-regulation of learning and performance: Issues of educational applications*, pages 44–73. Hillsdale, NJ: Erlbaum.

Bowen, C. W. (2000). A Quantitative Literature Review of Cooperative Learning Effects on High School and College Chemistry Achievement. *Journal of Chemical Education*, 77(1):116.

Clark, R. C., Nguyen, F., and Sweller, J. (2011). *Efficiency in learning: Evidence-based guidelines to manage cognitive load*. John Wiley & Sons.

Entwistle, N. and Ramsden, P. (1983). *Understanding student learning*. Beckenham: Croom Helm.

Felder, R. M. and Brent, R. (2005). Understanding student differences. *Journal of Engineering Education*, 94(1):57–72.

Foldnes, N. (2016). The flipped classroom and cooperative learning: Evidence from a randomised experiment. *Active Learning in Higher Education*, 17(1):39–49.

Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H., and Wenderoth, M. P. (2014). Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences*, 111(23):8410–8415.

Gynnild, V. (2001). *Læringsorientert eller eksamensfokusert? Nærstudier av pedagogisk utviklingsarbeid i sivilingeniørstudiet*. PhD thesis, NTNU.

Gynnild, V., Holstad, A., and Myrhaug, D. (2007). Teaching as coaching: A case study of awareness and learning in engineering education. *International Journal of Science Education*, 29(1):1–17.

Gynnild, V., Holstad, A., and Myrhaug, D. (2008). Identifying and promoting self-regulated learning in higher education: roles and responsibilities of student tutors. *Mentoring & Tutoring: Partnership in Learning*, 16(2):147–161.

Hattie, J. and Goveia, I. C. (2013). *Synlig læring: et sammendrag av mer enn 800 metaanalyser av skoleprestasjoner*. Cappelen Damm akademisk.

Johnson, D., R., J., and Smith, K. (1998). *Active Learning: Cooperation in the College Classroom*. Interaction Book Co., Edina, MN, 2nd ed. edition.

Lage, M. J., Platt, G. J., and Treglia, M. (2000). Inverting the classroom: A gateway to creating an inclusive learning environment. *The Journal of Economic Education*, 31(1):30–43.

Lan, W. (1996). The effects of self-monitoring on students' course performance, use of learning strategies, attitude, self-judgment ability, and knowledge representation. *Journal of Experimental Education*, 64(2):101–116.

Marshall, D., Summers, M., and Woolnough, B. (1999). Students' conceptions of learning in an engineering context. *Higher Education*, 38(3):291–309.

Marton, F. (1981). Phenomenography — Describing conceptions of the world around us. *Instructional Science*, 10:177–200.

Marton, F. and Booth, S. (1997). *Learning and Awareness*. Mahwaw, NJ: Lawrence Erlbaum.

Negnevitsky, M. (2005). *Artificial Intelligence: A Guide to Intelligent Systems*. Addison Wesley, 2nd edition.

Osen, O. L. and Bye, R. T. (2017). Reflections on Teaching Electrical and Computer Engineering Courses at the Bachelor Level. In *Proceedings of the 9th International Conference on Computer Supported Education (CSEDU '17)*. INSTICC, SCITEPRESS. Paper accepted for publication.

Piaget, J. (1968). Six psychological studies. Trans. A. Tenzer.

Prince, M. J. (2004). Does active learning work? A review of the research. *Journal of Engineering Education*, 93(3):223–231.

Prosser, M. and Trigwell, K. (1999). *Understanding learning and teaching: The experience in higher education*. Buckingham: Society for Research in Higher Education and/Open University Press.

Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Pearson: Upper Saddle River, New Jersey, 3rd (international) edition.

Russell, S. and Norvig, P. (2013). *Artificial Intelligence: A Modern Approach: Pearson New International Edition*. Pearson: Upper Saddle River, New Jersey, 3rd edition.

Schaathun, H. G. and Schaathun, W. A. (2016). Learning mathematics through classroom interaction. In *The 18th SEFI Mathematics Working Group seminar on Mathematics in Engineering Education*, pages 155–161.

Schroeder, C., Scott, T. P., Tolson, H., Huang, T.-Y., and Lee, Y.-H. (2007). A Meta-Analysis of National Research: Effects of Teaching Strategies on Student Achievement in Science in the United States. *Journal of Research in Science Teaching*, 44(10):1436–1460.

Sotto, E. (2007). *When teaching becomes learning: A theory and practice of teaching*. Bloomsbury Publishing.

Springer, L., Stanne, M., and Donovan, S. (1999). Effects of Small- Group Learning on Undergraduates in Science, Mathematics, Engineer- ing and Technology: A Meta-Analysis. *Review of Educational Research*, 69(1):21–52.

Topping, K. and Ehly, S. (1998). *Peer-assisted learning*. Routledge.

Topping, K. J. (1996). The effectiveness of peer tutoring in further and higher education: A typology and review of the literature. *Higher education*, 32(3):321–345.

Vygotsky, L. (1978). *Mind in society*. London: Harvard University Press.