Universiteit Antwerpen

Assignment Persuasive Technologies & Distributed Artificial Intelligence

Lauren Coppens, Kasper Engelen, Franciscus Fekkes, Geert Goemaere, Hanne Maes

1 The target group and their consumer needs.

Getting a driver's license and learning how to drive a car can be a challenge for both the student and the student's parents. The student can't learn everything during driving lessons with a professional instructor. There are a lot of things the student has to practice alone or with another non-professional instructor (like their parents). Parents aren't professional driving instructors, so they often don't know the exact steps of the learning process. Many students fail the first time they do the driving test. With our application we offer a solution to this problem.

There are two important groups involved in the problem: students who learn how to drive, and the non-professional instructors who teach the student how to drive. As a student (in Belgium) you can drive and practice alone (without an instructor) if you have taken 20 hours of driving lessons. But how do you practice alone? Which skills do you need to develop better in order to pass the driving test? Are you practicing everything the right way? And in what order should you practice all the skills? Which skill do you need to learn first in order to be able to master another (harder) skill? If you practice alone, there's nobody in the car with you to give you some tips or to tell you what you should pay attention to.

The application can also be useful for the second group involved in this problem: the instructors. In Belgium a non-professional instructor (for example: your parents) can teach the student how to drive a car. As a non-professional instructor it can be difficult to remember all the necessary steps in the learning process. The instructor already drives a car for years, and most of the skills have become a habit.

Our application will solve the problem of both groups involved. This application will be your virtual assistant while learning how to drive a car and will give exercises that the student can practice. When the student masters the new skill, he can indicate this in the application. That way you can practice every skill until the student reaches the final goal: mastering all the driving skills at such a level you are ready to pass the driving test.

2 How can reinforcement learning be implied to solve the problem?

Consider a student driver having to master N driving skills within a number of learning steps T. The app's goal is to maximize the number of skills acquired at the end of the training process. Suppose $x = (x_1, x_2, \ldots, x_N)$ is the student's skill set with $x_i = 0$ if the i-th skill is not yet acquired or $x_i = 1$ if the skill is already acquired. At the beginning of each step t, the app recommends a driving exercise a, from a catalog of learning exercises. By executing the driving exercise, the student's skill set x_t transfers to x_{t+1}. The student is rewarded with the skills acquired at the end of the exercise, so reward $r_{t+1} = |x_{t+1}| - |x_t|$. By recommending a sequence of exercises $a_1, \ldots, a_T$, the goal is to maximize the expected total number of skills during the learning process. The next action or exercise depends on what's already learned. The user will report which skills have been obtained (x_{t+1}) at the end of the exercise. There is some uncertainty in this assessment. Since the skill profile is an estimate, the following reward signals are estimates as well. Also the effectiveness of the exercises is possibly inconsistent and dependent on the situation and on the student. So increasing the skills transitioning from x_t to x_{t+1} is not guaranteed. We assume that once a skill is acquired, the student doesn't lose that skill. This assumption is acceptable for a driving course that happens in a relatively short time span.

2.1 State space

The state is a tuple of N skills, each having a value of 0 (skill not acquired) or 1 (skill acquired). Therefore, each state looks like $S = (x_{1,} x_{2,} \dots, x_N)$ with $x_i \in \{0,1\}$ in state space $\mathbf{S} = \{0,1\}^N$ of size 2^N, with N the different skills to acquire. Start state S_0 is set to the student skill set at the start of taking the driving course. For rookie students, this will be $S_0 = (x_{1,} x_{2,} \dots, x_N)$ with $x_i = 0, \forall i \in [1, N]$

<u>In our control:</u>
We have full oversight over the state space. All possible states are known beforehand. All possible skill levels to acquire and to succeed to complete the driver learning trajectory are known upfront. The initial state depends on the proficiency a starting student already has, which can be entered in the app's user profile.
<u>Outside of our control:</u>
The user has to manually specify how proficient they are at different driving-related skills at the end of each driving exercise. This is an estimate and has uncertainty in it.

2.2 Action space

An action is a driving exercise recommended by the app and chosen by the student. The driving school providing the app to the student has a catalogue of driving exercises $\{a_1, a_2, \dots a_C\}$ of size C. So, the action space contains C actions. A specific action (driving exercise) is constructed to improve one or more specific skills (e.g. parking skill). So, action space A= $a_1, a_2, \dots, a_C$ of size C.
<u>In our control:</u>
The app recommends an action (driving exercise) to the user.
<u>Outside of our control:</u>
The choice the user makes out of the recommended list of actions.

2.3 Reward signal

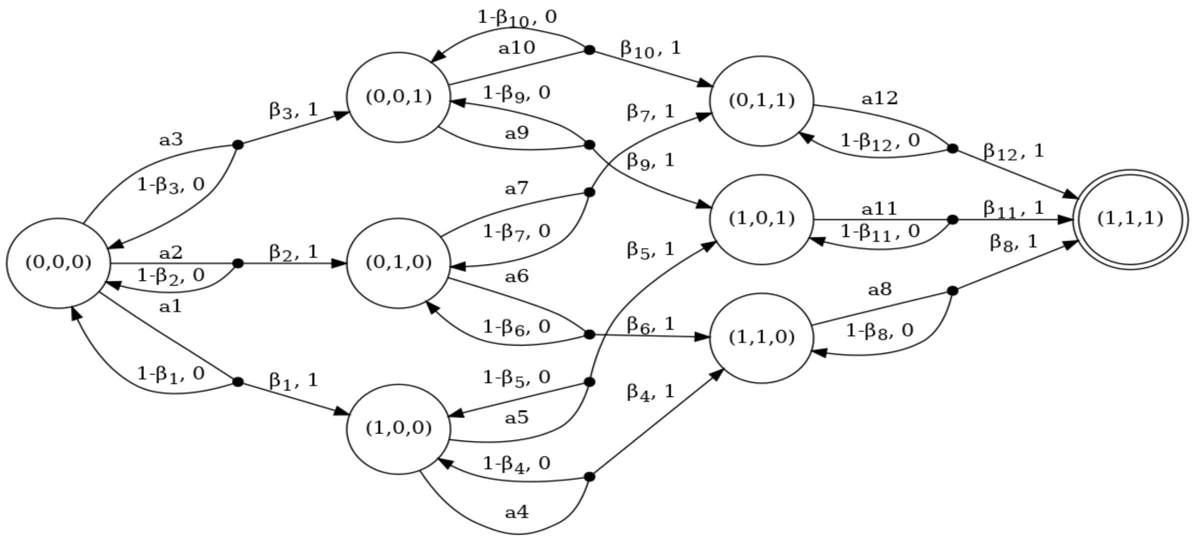
The reward is the signal provided by the user of the app. After performing each exercise, the user specifies whether the exercise was successful or not. The value of the reward function is the number of acquired skills. i.e. if two skills were acquired after an exercise, the reward function is +2. If no skills were acquired the reward is 0. So, the immediate reward is:

$$R_{t+1} = \sum_{k=1}^{N} [x_k(t+1) - x_k(t)]$$

<u>In our control:</u>
The reward signal can be calculated at the end of a driving exercise. i.e. when an action is taken in a specific state. When the exercise is assessed by the user, we can calculate the immediate reward based on the formula above.
<u>Outside of our control:</u>
The user has to manually specify how proficient they are at different driving-related skills at the end of each driving exercise. This is an estimate and has uncertainty in it.

2.4 The dynamics function

As we don't know the learning model of a student, and there's a lot of uncertainty, we will use online Q-learning to find the best policy (set of actions) for a learning process. The app will suggest the action (i.e. exercise) with the highest q-value in the current state. After performing the exercise, the user can report to the app which skills were acquired during the exercise (see reward function). The app will then transition to the next state based on the total set of skills that were acquired. However the basic structure of the model will be known beforehand (see diagram).



In the MDP diagram, an example is shown for N=3, resulting in 8 states. In reality the number of skills (N) will be larger and result in a finite state space of 2^N states. In each state, an action a_i (exercise) will be recommended according to a policy distribution. With some probability $\beta_i$the action succeeds it fails with probability $1 - \beta_i$.

<u>In our control:</u>
The state space, action space and reward signals are well defined.
<u>Outside of our control:</u>
We don't have precise knowledge on the learning trajectory of the student or any others students. Also the assessment of the student gives inaccurate information. Therefore the recommendation policy will be stochastic.

3 Human AI interaction paths and ethical concerns of the application.

According to Sundar and his Theory of Interactive Effects (or TIME Model), there are two specific routes that need to be taken into account when developing an application that is based on AI. The first route is the cue route. This route specifies how people perceive AI. For our application, AI could be seen as something helpful, accurate, neutral and objective. As explained above our application gives the student driving exercises to improve his or her driving skills. This way they can get ready for their driving exam. But you still need to fill in your own feedback. The application trusts your own judgement or the judgement of the non-professional driving instructor. Based on this information it gives the user extra exercises. On the negative side, this AI could be seen as something cold. You give the AI information about your driving skills. The AI processes the information and gives you exercises and a place where you can practice. This is a very cold approach. It’s important to change the mindset of potential users. They need to know our AI isn’t cold hearted. Our AI would be able to give recommendations, tips and tricks and support. When it recognizes your problems with a certain skill, it will give a mental boost. This way you will not give up and keep on trying. The AI would like to give you a positive feeling about driving a car. The AI wants the student driver to receive their driving license. People need to get the idea that the AI really wants to help.

Also, this sort of application is quite new. People don’t have anything to compare this application to. Many people are still hesitant to replace a human being with AI. Especially when they don’t know this sort of AI. This could be a problem in the beginning. We need to make sure that people trust the AI in the application before making any further technical developments. To ensure this trust, we would like to be open about our application. We would tell them what our AI needs to know to give the right exercises. In the privacy statement of the app, we would make clear which kind of data we keep and what we do with it. Transparency is important and can lead to better user engagement. We would also explain why we developed the application. We all had to learn how to drive. We all know how exciting and at the same time how frustrating it was to be behind the wheel the first time. And not only for the student drivers, also the driving instructors. We want to give the potential users the feeling they are not alone. The characteristics of the development, the reason why, can shape the perception of the application. This needs to be researched more deeply.

Our AI is also visible in the application. In the past you had this paperclip in Word where you could ask questions to and the paperclip would answer. In that particular case, the AI gets a face. It makes it visible and transparent. It’s not just a machine that answers. The characteristics of the look needs to be researched. Not every characteristic is appealing for everyone. Because the application is quite new, we need to stay vigilant about the users perceptions. Users don’t have prior experiences to build on.

We need to make clear that this application is an extra help. It cannot fully replace a human drivers assistant. Certainly not in the beginning. It gives options to get better at driving, but it still needs a human hand to make it work. We do not want young drivers to get behind the wheel for the first time without a supervisor or they can’t fill in feedback while driving. That would be really dangerous. It would also affect the perception of our application in a very negative way. That is why we use precaution messages. An example of a precaution message could be: “Do not enter data while driving. Using a phone while driving is very dangerous.” More in depth research is necessary to find out what messages are effective.

The second route from the TIME model is the ‘Action route’. This route emphasizes what kind of actions technology makes possible. The action route is divided into four variables. The first two variables are interaction and control. Our AI system needs input to work. Users need to mention how experienced they are in driving or the fact that the exercises went well. According to that feedback the app will give you exercises. If you said you succeeded in parking, but in reality you failed miserably, the application is still going to the next skill and that won’t do you any good. So when you give honest feedback to the application, it could be a really helpful assistant driver.

It also gives the user the perception of having control over the app. Because of the info they put into the application, they receive a different exercise on the same skill or move on to the next skill. Users also can control the way our AI looks and sounds like. Some people might like to hear a woman's voice or they prefer a different language. But the user does not have all the control. In our application we have a skillset the user can obtain. But it isn't useful when only the user can choose a certain skill. For example, a student driver likes parking and keeps choosing exercises on parking. They mentioned they failed the exercise. When the user has all the control, he won't learn the other skills. So we keep a little control of our AI. The AI will make sure there is enough variety between skills and exercises.

The third variable is 'Costs and benefits'. People need to benefit from using AI and avoid hurdles. The cost needs to be as low as possible. So our AI doesn't need much information to start with. It needs the students training level, the skillset and the date when you would like to finish the training. Then put the GPS on your phone and start driving. The app would be really easy to use, straightforward and no unnecessary questions. The app can give notifications as a reminder the student still needs to train.

The last variable in the Action route is 'Human – AI synergy'. Both humans and AI need to get better. The user needs to get better at driving and obtain necessary skills. The AI needs to optimize his process. With the feedback from the user, the AI learns which exercises worked and which don't work. It evaluates his own suggestions. If the AI notices that a user is having trouble with a certain parking skill, it gives other options. So the user can evolve step by step, until it is ready for the smallest parking spot.

The AI evolves because it learns what people need to evolve. After completing all the different training, the student driver will be ready for his or her driving exam and the AI has learned to give better options for the next driver. All the mentioned factors above (symbol, transparency, the goal, user interaction, agency and mutual augmentation) need to establish trust in the AI.

When working with AI, there are four ethical concerns that need to be taken into account.
The first one is fairness. We think of our app as fair. There is no difference in gender, age or what kind of car they drive. Everyone will receive the same exercises in time. Our app will also be available for everyone who wants to learn how to drive. It's important to us that everyone who wants to drive a car has a certain skill level and is an experienced driver. It would be unfair to make the app available only for the upper class of society. In order to make sure we exclude no one, it's important to talk with our different stakeholders.

The second concern is about explainability. In the app we explain the technical processes and the effect of the user's feedback. But we don't want to give away too much. We explain why we give an exercise to a student and why this exercise is important. Also, as explained above we make our purpose and goal clear in a preserved part of the app. We want our app to be as transparent as possible. Because we think our app will be used by a variety of people, we would use simple language. That way we can make sure everyone understands the benefits and the risks of using the app.

Privacy is another ethical AI concern. Our app knows when you are not home. You need to do the exercises on the road. So it is very important that our system is safe from hackers. Users need to know what data we collect and what we do with it. Users can check their data at all times. Therefore it is necessary to have a privacy statement.

Finally, we have accountability as the last ethical AI concern. We need to make users aware of any dangers that come from using the app. As explained above, we have warning messages incorporated in the app. Drivers need their eyes on the road and not on the app. Also the users need to know who is responsible when an accident occurs when using the app.

4 RL approach.

To implement the recommendation system, we suggest using a full reinforcement learning technique called Q-learning, which is part of the "temporal difference learning" family of techniques.

As said before, every skill set would be a state s and every exercise would be represented by an action a in said state. Every q-value $Q(s,a)$ would represent the value of doing the exercise associated with the action a with the given skillset s. The suggested exercise(s) would correspond to the action(s) with the highest q-values. It is also possible to apply a exploration function f so that every exercise will be tested over time:

$$Q(s,a) += \alpha(R(s,a,s') + \gamma \max_{a'}(f(Q(s',a'),N(s',a'))) - Q(s,a))$$

$$f(u,n) = u + k/n$$

Where $N(s',a')$ is a function to indicate how many actions ago this action a', was taken in the state s'. This should give a low value to actions that were taken in the near past. $f(u,n)$ is a tunable function where k can be used to determine the influence of n. The influence of n should lower when the q-value table converges. Both f and Ncan be tuned/chosen to improve the learning algorithm.

To learn the q-values we first thought of using value iteration in combination with a database of past experiences from which we would derive a $p(s',r \mid s,a)$ function. Later it came to light that such a database would take time to create and it would be stationary. So instead an online approach would be more appropriate.

We eventually chose temporal difference learning, in which every experience could be used to refine the q-values. Of course we would need some initial values to jump start this process. Several options would come to mind. Either we could start with flat values for every (state, action)-pair or we could use expert data by asking instructors to rank each exercise and skillset combination. These values would not be good to use initially so a closed-beta would be recommended to get decent q-values before launch. These values would continue to be improved after the launch by the online Q-learning algorithm.

We would need an online database to collect and process the data. The user could have an app on his phone to ask for good exercises for his student. The app could store the experiences (state, action, reward, next state) until a suitable moment to update the app has occurred. The app would send its experiences to the server and retrieve an updated q-value table. The q-learning algorithm would run on the server.