**Presentation script**

**Intro Slide:**

* Hi everyone, welcome to my final project presentation titled Learning Shortcuts in the Chemical Space supervised by Alessandro Lunghi.

**Slide 1:**

* So, since that title is a bit nondescript, I thought it best to nail down exactly what the outline and objective of the project is. So the goal of the project is to leverage machine learning methods, in particular reinforcement learning, to efficiently screen a large number of molecules at low computational cost for promising single-molecule magnet candidates. So of course I need to discuss a little bit of background about reinforcement learning and firstly single-molecule magnets.

**Slide 2:**

* So, what are single-molecule magnets? Single-molecule magnets are molecules that show a magnetic memory effect, i.e. they retain their magnetisation at low temperatures. What I mean by retain their magnetisation is that they display a phenomenon known as magnetic hysteresis as seen in this diagram from the first discovered SMM Mn12Ac in 1993.
* When a positive external magnetic field is applied the sample becomes positively magnetised and if we reduce the field back to 0 the magnetisation is retained, it even requires a negative magnetic field to reverse the polarization of the magnetisation and we form a hysteresis loop.
* So in the 0 field condition we can have two possible states with the magnetic moments aligned in opposite direction respectively, in other words up and down or 0 and 1 states and we can store information just like a bit.
* When I say low temperatures, you can see here that for Mn12 this is only a few kelvin and the current best performing SMMs are effective just above the temperature of liquid nitrogen approx. 80K. We want to find SMMs that function at higher temperatures with the ultimate goal of reaching room temperature.
* Why does this happen? Due to (axial) magnetic anisotropy there is an energetically favourable direction of magnetisation known as the easy axis. The energy levels are represented by where D is the anisotropy parameter. And in particular for SMMs we require negative such that the ground state energy levels have the highest magnetic moments leading to a bistable ground state.
* So again, in this diagram we initially have no magnetisation, and we have even distribution in the ground states (equilibrium). Then we turn on the field, we get magnetisation in the energetically favourable direction and this is retained when we turn off the field, and information is preserved.
* So you can see depicted here that to return to equilibrium the system needs to overcome an energy barrier or equivalently be in an ambient temperature greater than what we call the blocking temperature. This is known as relaxation and we want to avoid this at all cost because we lose the information!
* In reality at finite temperature there is a finite probability of relaxation to occur given by the relaxation rate due to spin-phonon coupling: , which we can see depends on both U and T. We want to make this rate as small as possible so we have two options. 1 decrease the temperature, expensive and not practical… so we are left with increasing the energy barrier.
* And so we want highly negative anisotropy values!

**Slide 2.5:**

* So, why are SMMs important? We could make memory storage devices such as hard drives a factor of 100 smaller! Also applications for quantum computing as SMMs can maintain quantum states for a relatively long period of time they could serve as stable qubits

**Slide 3:**

* Now for a short discussion about reinforcement learning, often abbreviated as RL. So, reinforcement learning is one of the three main machine learning paradigms along with supervised and unsupervised learning.
* However it is fundamentally different these other forms of machine learning as it does not require labelled training data, supervision or explicit correction of sub-optimal actions.
* Instead an RL agent learns through trial and error by interacting with its environment and receiving rewards or penalties based on its actions. In other words we reinforce positive behaviour and punish undesirable behaviour.
* I think this is very intuitive and natural for us as humans. It the same idea as how we were taught to behave as children. You behave well and eat all your vegetables and you get ice cream after dinner, or you misbehave and get put on the naughty step.
* Another important thing to understand in RL is that feedback is delayed and an agents actions can influence the subsequent situations it finds itself in, its not just about maximising immediate reward but also taking the future consequences of your actions into consideration.
* For example maybe you steal €100 from the shop so you get a large immediate reward but as a result of that action a few days later you get found out, arrested and throw in jail as a result. Similarly, maybe you turn down some immediate reward like a 25k salary job straight out of secondary school and instead you pay to go to college and get a degree to eventually get a better paying job and make more money in the long run.
* Also important is considering the balance of exploration and exploitation. After some time the agent has built up some knowledge of its environment and the consequences of its actions. Exploitation involves taking ‘greedy’ actions with the best future reward from current knowledge. However, exploratory actions without reward in mind are also important to discover potentially greater rewards from new situations.
* RL is based upon the reward hypothesis which states that all goals can be represented by the maximisation of expected cumulative reward.
* RL is extremely versatile in dealing with complex decision making processes. Some of its biggest contributions have been in: Finance, robotics, autonomous vehicles, game playing etc. Money is of course quite a natural choice for reward.
* In particular, one of the most significant milestones of reinforcement learning has been AlphaGo developed by Google DeepMind in 2016 that defeated the best human Go player of all time over a decade earlier than experts predicted. Go is an extremely complex game with an average of about 250 possible moves per turn, so classical ML search methods would be futile. (image)

**Slide 4:**

* So, now we will briefly discuss Markov Decision Processes or MDPs, a formalization of sequential decision-making processes that provide a mathematical framework for RL problems.
* So this diagram depicts the agent-environment interface in an MDP, the agent is simply the learner and decision maker and the environment comprises everything outside of the agent.
* At each time step the configuration of the environment is represented in state . The agent takes action based on . And the environment moves to a new state due to this action. In general this evolution to a new state can be deterministic in that an action will always lead to the same next state, or stochastic with a distribution of possible next states due to the action each with some probability of arising. The agent then receives some numerical reward and is presented with the new state representation and this repeats until a terminal state is reached.
* In particular, all the states in an MDP are said to have the Markov property, which basically means that that the current state is a sufficient description of the full history. So, this is what it means to have the Markov property, the conditional probability distribution of future states depends only on the current state. . Give example

**Slide 5:**

* So what determines the behaviour of the agent, how does it choose what action to take? The agent follows a policy pi which is a mapping of states to actions. So this denotes the probability of taking action a given that we are in states s.
* We also need something to evaluate the performance of a given policy. A value function gives a prediction of future reward while following a certain policy. It is used to evaluate how good/bad it is to be in a particular state. So here is the action-value function which describes given that we start in state s and take action a what is the expectation of cumulative reward while following a particular policy pi. These gamma factors are known as discount factors between 0 and 1 meaning we weight future rewards progressively less and less. This is a user defined parameter that can essentially give more priority to immediate rewards. This is useful in a number of ways but it mainly represents future uncertainty and is also convenient to ensure convergence and stability of RL algorithms.
* So how do we find the *optimal* policy of our agent, how do we find what the best possible behaviour is? Well this is actually just equivalent to finding the optimal action-value function as then the optimal policy is simply acting greedily with respect to it. All this says is that in state s we always choose the action that maximises the optimal action-value function, we simply pick the action that gives the highest value.

**Slide 5:**

* So how do we determine the optimal action-value function ? For this project we mainly used a well known algorithm known as Q-Learning where capital Q just represents the learned action value function.
* It is a tabular method meaning that we generate a table like this with rows representing states and columns representing actions and we populate the table with the learned value function of these state action pairs.
* It is also an off-policy algorithm meaning that we learn the optimal policy indirectly through a behaviour policy. So our target policy isn’t actually used in the algorithm it just learns from the behaviour policy.
* It is also a temporal difference algorithm which just means that after we take a certain number of steps we approximate the value from that point onward with our current knowledge, this is known as bootstrapping. As a pose to having to take a full trajectory through the MDP before updating as in Monte Carlo methods.
* The behaviour policy used is known as epsilon greedy in which we choose an epsilon in the range 0 to 1 and take an exploratory action with probability epsilon and otherwise take the best action from our current knowledge.
* So here is the Q learning algorithm in all its glory. We update the Q value of a state action pair each time we take a step of our epsilon greedy policy. We then move into a new state S(t+1) and add the received reward plus the maximum value of being in our new state onwards, discounted. So we look ahead and see from our current knowledge what value to best action in out new state would give us, discount it and add to the reward. This is our TD target and represents what we what we want to move our Q value towards. We just subtract our current estimate giving us the TD error and move our Q value a small step alpha in that direction.
* And indeed this method does converge to the optimal value function.

**Slide 6:**

* So what exactly is our problem and how will we apply these reinforcement learning techniques to solve it? We construct molecules by adding ligands to a central metal atom, in this case cobalt.
* A ligand is simply a molecule or ion that binds to a central metal atom to form a coordination complex.
* So our actions are the addition of each ligand, in this case a carbonyl, we receive a reward of 0 and then our new state in just the entire construction so far so of course it has the Markov property and this is a valid MDP. We continue until we add the final ligand of however large a molecule we have decided and then we receive a final reward of minus the anisotropy value because we want to maximise for -D.
* This full process constitutes one episode of the MDP and we repeat many time to learn via Q learning
* We want to efficiently find the most highly negative anisotropic molecules, we choose what ligands to add via the epsilon greedy policy.
* This is a very challenging Markov decision process, it is extremely short and we get no information until the episode terminates. Due to this we can set gamma and alpha both to 1 (alpha because deterministic state transition dynamic). (Essentially the algorithm simplifies to just replacing with the best value of the next state)
* The only data that we provide is simply the molecule name and the corresponding anisotropy value.

**Slide 7:**

* First we wanted to investigate if this is even a reasonable approach to solving this problem so we began with Synthetic Data in almost like a sandbox box where we could easily vary the number of ligands we can select from, N, and the size or total number of ligands in a molecule M.
* We then have a total number of states including intermediate states given as:

When choosing with replacement. And here is a table representing how to number of states grows with M and N.

* We start with the most simple case. We assign a number in [-10, 10] to each ligand and when we reach the end of an episode/complete the construction of a molecule we simply sum the value of each ligand in the molecule and take this as the reward so there is no correlation between the ligands whatsoever.
* Naturally the optimal molecule will simply be the ligand with greatest value repeated M times.
* We also define a performance ratio which is the total number of terminal states visited divided by the total possible no. of terminal states. Terminal state just means a constructed molecule. This gives a measure of how efficiently our algorithm finds the best molecule.
* We also define a cost ratio which is the total number of unique terminal states/ molecules divided by total possible number of terminal states. This represents how much of the state space we had to explore before finding optimal molecule and this is really what we want to minimise as much as possible
* So what does finding the best molecule typically look like?

**Slide 7:**

* We realised we could make some improvements to the vanilla Q learning algorithm due to the deterministic nature and the problem and order not mattering when adding ligands. Instead of just one update per Q learning step we can simultaneously update any state action pair that could have led to the same state: See diagram
* **Spintronics – transfer info**
* **Spin relaxaation**
* **Phonons**
* **Solve problem energy barrrier**

New applications slide:

* So what are SMMs? They are molecules that show a magnetic memory effect, in fact they are the smallest material unit with memory effect. The problem is at the moment they are only effective at extremely low temperatures The current best performing SMMs are effective just above the temp of liquid nitrogen at around 80K. So we want to try to find SMMs that are effective at higher temperatures with the ultimate goal of above room temperature performance.
* So what are some of the potential applications of room temp SMMs?

A diagram of a field

Description automatically generated

**Relaxation**

*+S*

*-S*