universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

'Sensors and Triggers in Social Energy Systems"

verfasst von / submitted by

Franz Papst, BA BSc

angestrebter akademischer Grad / in partial fulfillment of the requirements for the degree of

Master of Science (MSc)

Wien, 2018 / Vienna 2018

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 066 013
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Joint Degree Programme MEi:CogSci Cognitive Science
Betreut von / Supervisor:	Ass.-Prof. Dr. Olga Saukh

Contents

Abstract

Generation and distribution of electrical energy is going through a rapid change, shifting from the traditional one-to-many relation, where one entity is in charge of generating and distributing energy, towards a many-to-many relation, where consumers are not just passively consuming energy, but are also able to generate energy and sell their surplus energy back to the power grid. This thesis is investigating two aspects of this transition: 1.) What are the driving factors behind this shift and how this new, decentralised and more democratic approach of energy generation can be exploited to increase profits for a small group of participants on the grid, while the majority of participants is facing negative consequences. I will use a game theoretic point of view and agent-based modelling to show this. 2.) Photovoltaic arrays are one of the main drivers behind this paradigm shift, yet it is not clear how many of them are installed and in use. I will propose a system to automatically detect solar panels from high resolution aerial images, which uses a deep-learning approach for object detection.

Zusammenfassung

Gewinnung und Vertrieb von elektrischer Energie befindet sich in einer Phase der Veränderung, weg vom traditionellen $1 : N$ Schema, wo Energie von Unternehmen erzeugt und vertrieben wird, hin zu einem $N : N$ Schema, wo Konsumenten Energie nicht mehr nur passiv konsumieren, sondern auch in der Lage sind Energie selbst zu erzeugen und überschüssige Energie zurück an das Netz zu verkaufen. Diese Arbeit beschäftigt sich mit zwei Aspekte dieses Wandels: 1.) Was sind die treibenden Kräfte dieser Veränderung und wie kann dieser neue, dezentralisierte und demokratischere Ansatz der Energiegewinnung ausgenutzt werden, um die Profite einiger weniger Netzteilnehmer zu steigern, während die Mehrheit der Netzteilnehmer den Preis dafür zahlen muss. Ich werde dies von einer spieltheoretischen Perspektive aus beleuchten und mit einem agentenbasierte Modell zeigen. 2.) Photovoltaikanlagen sind verantwortlich für diesen Paradigmenwechsel, jedoch ist nicht klar wie viele Anlagen existieren und in Betrieb sind. Ich werde ein System präsentieren, dass in der Lage ist Photovoltaikanlagen automatisch in hochauflösenden Luftaufnahmen zu erkennen, dieses System verwendet Deep-Learning Methoden zur Objekterkennung.

List of Tables

List of Figures

Chapter 1

Introduction

Energy generation and distribution have been for the longest time seen as technical challenges and problems. This was, and for the most parts still is, true, but changes in the underlying technology have lead energy generation to becoming more and more a social enterprise as well. Generating energy does no longer require large power plants, but it can also be achieved by much smaller facilities like photovoltaic arrays. This shift in technology drastically decreased the entry barrier for actively participating on the energy market. It nowadays allows a large group of people to generate energy on their own, instead of just relying on utility companies. One example for such a social energy system is the Brooklyn Microgrid [56], a community driven power grid located in Brooklyn. It allows participants for example to sell excess energy they generate from their solar panels to the grid. Participants are also able to choose their preferred energy sources, which has the potential to boost green and sustainable energy sources. The microgrid uses a blockchain to handle the trading amongst participants and to track the generated and consumed energy [95]. Decentralisation of energy systems has a lot of potential and advantages, but also some drawbacks. My thesis is split into two parts, which study social energy systems from different angles.

In chapter 2, I will further elaborate microgrids, their advantages and drawbacks as well as related concepts like the energy market or renewable energies and look at them from a game theoretic perspective. Furthermore, I will show a potential thread to the microgrid, where participants, who also generate energy, are able to drive up the price for energy on the microgrid to sell their energy for higher profits. I am going to use agent-based modelling to show that participants are able to learn this unfavourable behaviour on their own. This chapter is based on a paper I co-authored with supervisor Olga Saukh [75]. Compared to our publication, I will give more background information and in general provide more details about the work we did and our implementation in this chapter.

In chapter 3, I will introduce a system to automatically detect solar panels from high resolution aerial images. Due to falling acquisition cost, the adoption of solar panels is becoming more and more widespread, but at the same time it is difficult to track how much solar panels are installed and how this number changed during recent years, because there are no available databases containing this data. At the same time this data would be interesting for policymakers as well as for utility companies and related third parties. In this chapter I propose a system based on the *You Only Look Once (YOLO)* algorithm [72] for object detection. YOLO is a state-of-the-art algorithm for object detection and is fast enough in detecting objects to be used for object detection in videos, while still providing good enough detection performance to compete with other state-of-the-art approaches. In this chapter I will give a general introduction to image recognition and object detection, furthermore I will describe the architecture of the system and give details of the implementation as well as evaluating results from using this system on high resolution aerial data from Vienna.

This thesis takes an interdisciplinary approach towards studying the power grid, besides using methods from power engineering it also uses methods and approaches from game theory, artificial intelligence and economics.

Chapter 2

Synchronisation Games in P2P Energy Trading

2.1 Problem Definition

Energy generation and trading was and for the most parts still is a one-to-many relationship, where the power grid was owned by the same entity that also generated, sold and distributed the power. Due to the special nature of electrical energy[1] and the rather complex process of generating it, it seemed that different approaches are not possible, or are only possible in very limited environments. This notion is about to be change due to the rise of distributed energy generation technology. The network operator[2] is not the only entity that is able to produce energy any more. A so called *prosumer* is also able to generate energy to meet his own energy demands and can also sell a surplus of the generated energy to other consumers in the neighbourhood. Generation is mostly done by solar panels and the trading is preferably done, utilising distributed ledger technologies like public blockchains and therefore in a truly peer-to-peer fashion, cutting out all middlemen.

This change in the way how energy is produced and distributed creates the need for a change in the topology of the power grid. A modern power grid must be able to take peer-to-peer energy trading into account. Even though, there is a lot of research and development going on in this direction, the integration of this new technological (and to a certain degree societal) developments is challenging, since the power grid as it is currently in use, was never designed for

[1] Electrical energy can practically only be used at the same time it is generated. Producing electrical energy ahead is not really possible, since storing it is only possible with heavy limitations and does especially not work very well for large quantities of electrical energy.

[2] Also commonly referred to as grid operator.

consumers being able to sell energy as well. The biggest change is that the owner of the power grid is now not the only entity to generate, sell and distribute electrical energy over the grid any more. This has a large impact on the whole system, most obvious in the fact that prosumers and network operators have different interests: the network operator is mostly interested in keeping the power grid stable, a prosumer on the other hand cares less about the stability of the grid, but is more interested in maximising his profits and selling his surplus energy as expensive as possible. It is the network operator's duty to keep the network stable at all times, to fulfil this duty network operators try to balance the current need for electrical energy generated with the current demand. The consumption curve for one day follows roughly the same pattern every day: one large peak in the morning, when people wake up and turn on electrical appliances, which is followed by roughly 8 hours of steady but lower consumption and by another peak in the afternoon when people are coming home from work. These peaks are well-known to the network operators and energy generation is allocated according to those needs. In case that the load on the power grid is higher than predicted, network operators have additional backup power plants, the spinning reserve [36]. The base load of the grid is usually met with power plants that are rather cheap to operate and quite inflexible to turn on or off (like nuclear or coal power plants). Intermediate loads are usually met with hydroelectric power plants, very high peaks are met with turning on gas- or oil-fired power plants, which are very flexible when it comes to staring up and shutting down, but are very expensive to use, since they are not very efficient and use rather expensive fuels, so they are only used for extreme spikes and may only run for a few hours per year [36]. Using these reserves for generating energy comes with high costs for the network operator, which results in higher prices for the consumers. While higher prices are negative for consumers, they are beneficial for prosumers, since they allow them to sell the energy they generate for a higher price and thus making a higher profit. The price set by the network operator is the upper boundary for prosumers. Since consumers would not buy energy from a prosumer, if it is more expensive than buying it from the network operator. Another side effect is that activating the spinning reserve has a negative impact on the environment, since they heavily rely on burning fossil fuels or gas. This would pervert one of the ideas behind having prosumers: to tackle global decarbonisation[3].

This conflict of interests creates some friction, which I am going to further elaborate in this chapter. I will take a game-theoretic look at this problem and I will show that malicious prosumers can cooperate in order to artificially drive up the energy demand of the power grid and increase the energy price for all consumers of the grid, which leads to higher profits for the

[3]This topic will be further elaborated in sections 2.1.1 and 2.1.4.

prosumers. For the prosumers this behaviour will result in a socially-deficient Nash-Equilibrium, a state that maximises the utility for them, but at the same time this state would not maximise social welfare and have negative consequences for all other parties involved.

2.1.1 Renewable Energy Sources

The influence humans have on the global climate is well documented [38]. This knowledge lead to a change in global climate policies, especially recently, driven by the framework set by the 2015 United Nations Climate Change Conference in Paris [84], which pushes for a rapid global shift towards renewable energy sources to minimise the effects of climate change. This shift in energy policies and the slow depletion of fossil fuels lead to renewable energies getting more and more attention from different actors in the energy sector.

In parallel to the developments mentioned above, the costs for generating renewable energy were constantly decreasing over the last decades, while the efficiency was constantly increasing. Nowadays the costs of using renewable energy sources are approaching the costs of traditional energy sources and will likely become cheaper in the near future. Swanson's law for example suggest that due to better production methods the price of solar panels tends to decrease by 20% for every doubling of the totally produced amount of solar panels [78]. By extrapolating this learning curve, solar panels could provide the world's current power demand by 2032 [66].

From a technical point of view using renewable energy sources has some advantages, but also some challenges. One technical advantage of renewable energy sources is, that renewable energies can be generated much closer to where the end-user resides than most other means of energy generation[4], which results in a better power quality since voltage fluctuations and line losses are lower than from traditional power plants [8]. Another advantage is that a network which is generating energy from renewable sources is in general more distributed and decentralised, since the generation facilities do not require large power plants to work. This makes the power grid more redundant and therefore more resilient, since it removes single points of failure.

However, integrating renewable energies to the existing grid is a big challenge. Probably the biggest challenge is that unlike traditional energy sources, renewable energy sources are not able to constantly generate energy during a day. The amount of generated energy is also highly depended on current weather conditions and very difficult to predict. As mentioned above, due to the special nature of electrical power, this is a big drawback for renewable energy sources and probably also the reason why it took them a rather long time before they were widely adopted. Especially solar generated energy suffers from the fact, that the peak of the generation usually

[4] A solar panel can be mounted on a rooftop of a house, while a nuclear power plant for example is built within a large distance from residual areas.

occurs after the consumption peak in the morning and before the consumption peak in the afternoon, thus diametrically to when it would be needed the most. In order to tackle the issue of unreliable generation, network operators have to still rely on traditional power plants to keep the network stable at all times. Improvements in the storage capacity and longevity of batteries will also play a key role in stabilising the power grid when using renewable energy resources. Also, further increases in the number of electrical vehicles and therefore increases in the number of high-capacity batteries can be utilised to add further storage capacity to the grid [82].

2.1.2 Energy Market

Efforts to liberalise the energy market like the European Union's Third Energy Package [86] include measures like ownership unbundling and strengthening consumer's rights, like the right to change the electricity provider free of charge. Even though measures like these allow consumers to freely chose their electricity provider and should cater for more competition and thus lower prices for the consumers[5], the physical distribution of the electrical energy is still done via the grid the consumer is connected to and thus via the local electric utility company, the energy is also most likely generated by the local electric utility company. While there are practically no differences in terms of generation and distribution, different electrical energy providers trade energy on the wholesale market [61], where they bill each other and handle all the financial accounting [6], the consumer does not notice any of the processes going on in the background, he just pays his electrical energy provider.

While residual consumers have in general no possibility to participate in the energy market[7] and therefore have fixed prices, commercial and industrial consumers can have different contracts, like time-of-use pricing, where they get charged higher prices during times when the general load of the power grid is high and consequently lower prices, when the general load of the power grid is low [76].

A rather new approach for energy trading is peer-to-peer (P2P) trading, P2P energy trading

[5]While measures to deregulate the European energy market seem to be quite successful and yielded the expected results, in other parts of the world the deregulation of the energy market had disastrous results, like the California electricity crisis of 2000 and 2001 [37].

[6]E.g. consumer C is a customer of electricity provider 1, but connected to a grid owned by electricity provider 2, so provider 2 generates and delivers the energy and bills it to provider 1, which then bills the consumer.

[7]And in general they also have only very little possibilities to benefit from fluctuations in the demand of electrical energy and thus in fluctuations of the energy price. Demand-side response [25] is one way for consumers to adapt their energy consumption with respect to the current demand and therefore get compensated with lower energy costs. Demand-side response programs already exist [25], but they are still in very early stages and mostly focused on research and thus far from receiving mainstream adoption.

is defined as the transfer of energy from an entity that produces energy to an entity with an energy deficit within a certain time interval [67]. Since pure P2P energy trading is highly depended on supply and demand and therefore heavily limited when it comes to providing a stable grid[8], grids where P2P energy trading happens are connected to a larger grid that always fulfils the current demand therefore provides stable energy source. P2P energy trading gives the following four advantages to its users [75]:

1. The ability to personally decide what type of energy to buy.
2. The usage of energy is more local and thus the line loss [8] is reduced.
3. The whole system has a higher reliability due to more redundancies.
4. Renewable energy sources and energy storage solutions can be seamlessly integrated on the edge.

There is already a number of projects investigating P2P trading mechanisms, incorporating new technologies like blockchains and smart contracts [64] [48] [45] [30] [34] [77].

2.1.3 Microgrid

Renewable energy sources and the relative ease of generating them, also enables new ways of distributing and selling energy. Like previously mentioned, the emerging new structures of the energy market are taking that into account. A microgrid is a new topology for the power grid which distributes low-voltage energy from distributed energy resources, i.e. solar panels and storage devices, i.e. batteries, that is able to handle flexible loads. It can be connected to the main grid or operate independently [35]. Due to utilising information and communication technologies, a microgrid is much more efficient in balancing the resources and therefore much more efficient overall.

A microgrid is constituted of the following three actors, which are depicted in figure 2.1.

Consumer

Like in a traditional power grid, a consumer in a microgrid can only consume energy. Since a microgrid utilises a lot of communication technologies, a consumer can install a demand-side response system [25] to shift loads away from peak periods and thus decreasing the overall energy costs.

[8] Especially if the energy is generated from renewable resources, which are even less reliable and predictable and thus also not beneficial for a stable power grid.

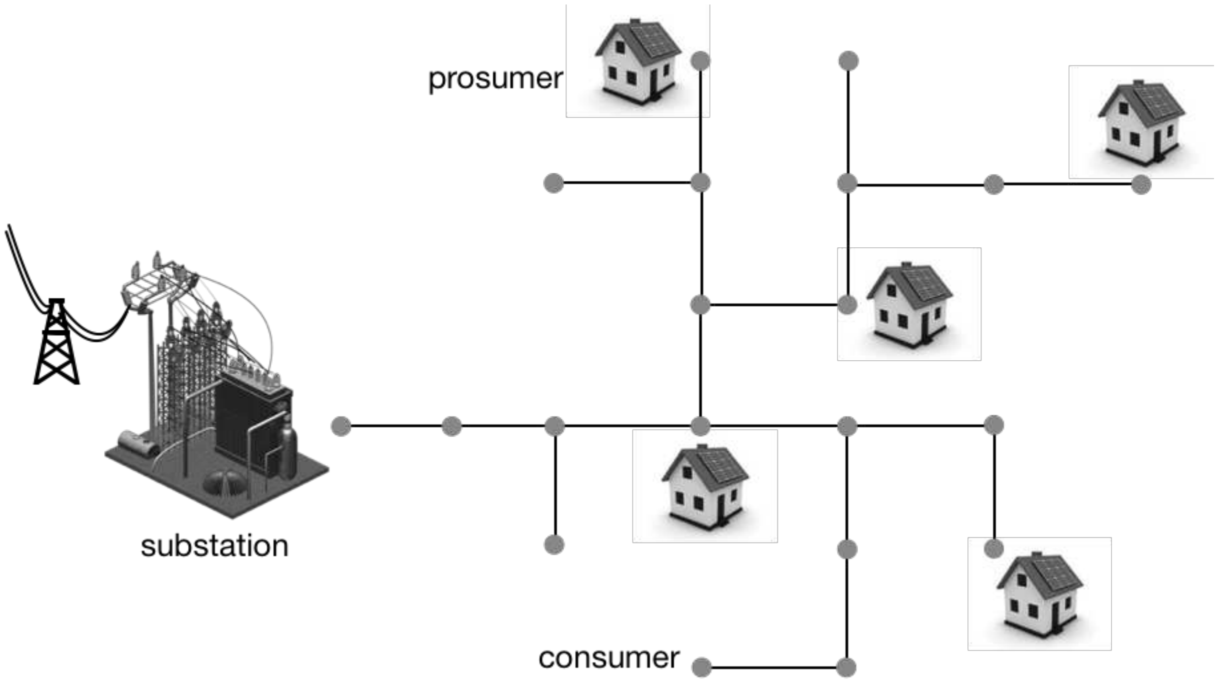


Figure 2.1: Schematic view of a microgrid. The microgrid is connected to the macrogrid of the network operator via a substation. The prosumers and consumers are all connected amongst each other via the microgrid. The consumers can only consume energy, either from the prosumer or the network operator, while the prosumer can supply themselves with energy and sell a potential surplus of generation to the consumers. If they are not able to fulfil their own energy demand, they also have to buy energy from the network operator.

Prosumer

A prosumer is an actor on the grid that consumes energy, but at the same time also produces energy and sells surplus energy on the microgrid. The energy generation is i.e. done by solar panels on the roof of the prosumer's house. The more active role a prosumer has on the grid, due to the capability to generate and sell energy, leads to new opportunities that have not been possible before, like synchronisation games, that will be further elaborated in section 2.1.4.

The term prosumer was coined in 1980 by Alvin Toffler [81] to describe how the role of consumers is changing in a post-industrial age. The blending of consumers and producers can be seen in different areas today, especially in social media, where most of the content is generated by the users of a platform.

Network Operator

Having a network operator is optional for a microgrid, but when a network operator is present, the stability of the microgrid is drastically improved, since the job of a network operator is to balance the load and keep the network stable and running at all times.

In most current power grids however, the same entity that generates energy also controls the grid for distributing it. In a microgrid this is not the case any more, since also the prosumers can use the grid for distributing power. This new paradigm can cause some friction and could lead to undesirable results, as I am going to elaborate in the next section.

2.1.4 Synchronisation Games

Network operators and prosumers are using the same grid, but they do have conflicting interests. A network operator for example is more interested in having a network that is stable at all times, while a (selfish) prosumer is less concerned with network stability and more with maximising his profits. These conflicting interests get even more apparent, when realising that the physical distribution of the power and the energy trading are done on two different layers, which are only loosely coupled with each other and therefore manipulations on one layer can have a big impact on the other layer. Since the network layer, where the physical energy distribution happens is rather difficult to manipulate, the manipulation usually happens on the trading layer.

To play a synchronisation game, a group of prosumers has to act together to artificially create a higher demand, thus forcing the network operator to activate the spinning reserve. Activating the spinning reserve comes with a high cost for the network operator, in addition to the activation costs, the fuels for the spinning reserve (like oil or gas) are rather expensive as well. The higher costs for the network operator are then reflected in a higher electricity costs for the consumers[9]. Prosumers are interested that the network operator increases his price, since they generate their energy on their own and therefore are not depended on a low energy price to keep their energy costs low, but at the same time they are selling their surplus energy, where the price of the network operator serves as upper boundary, since consumers are interested in lower prices, because they are forced to buy energy from the market and therefore try to minimise their energy costs by buying the energy as cheap as possible. If the price of a prosumer would be higher than the price of the network operator, the consumers would always buy the energy from the network operator instead of the prosumer. So it is in the prosumers' best interest that the price of the network operator is as high as possible.

Figure 2.2 shows an example for a synchronisation game. Consider a microgrid where all prosumers have solar panels on their roofs on a sunny day. The consumers buy energy from the prosumers and the microgrid operates mostly independent of the main grid. A consumer has a certain demand d_t, which he buys on the market. A prosumer can generate enough energy to meet his own demand d_t and either sell the surplus to the market (p_t) or store it in his local storage (q_t). The profit the prosumer makes is $p_t \cdot s_t^p$, where s_t^p is the price set by the prosumer. If the overall generation in the grid is high, the market price for energy s_t is low[10], thus also the price s_t^p set by the prosumer is low. In order to be able to sell their generated energy for a higher price, the prosumers have to influence the market price s_t in such a way that it increases. This is done by a group of prosumers synchronising their actions and artificially creating a higher demand on the market, which, from the view of the network operator, corresponds to higher demand on the grid. In order to do this, one prosumer splits his current demand d_t into two parts, so that $d_t = d_t^{(1)} + d_t^{(2)}$. The prosumer now buys the first part of the demand $d_t^{(1)}$ from the market, while at the same time he generates this demand and puts it into his local storage. In the next step the prosumer normally generates his demand $d_t^{(2)}$ and sells a_t from his local storage. If there is still a surplus of energy left, it gets again either stored in the local storage or sold on the market, just like a "friendly" prosumer would do. For the prosumer this is pretty much a zero-sum game, given by

$$d_t^{(1)} \cdot s_t - a_t \cdot s_t^p \approx 0 \tag{2.1}$$

The prosumer has to minimise equation 2.1, ideally it is zero, but small losses can be toler-

[9]To illustrate the concept of synchronisation games, I am going to assume that the energy price in the microgrid is set dynamically and that all the energy trading is done P2P. So there are no fixed rates a consumer has to pay to the network operator, everything is billed depending on the current consumption of a consumer and the current market situation.

[10]In general the price is set by the network operator.

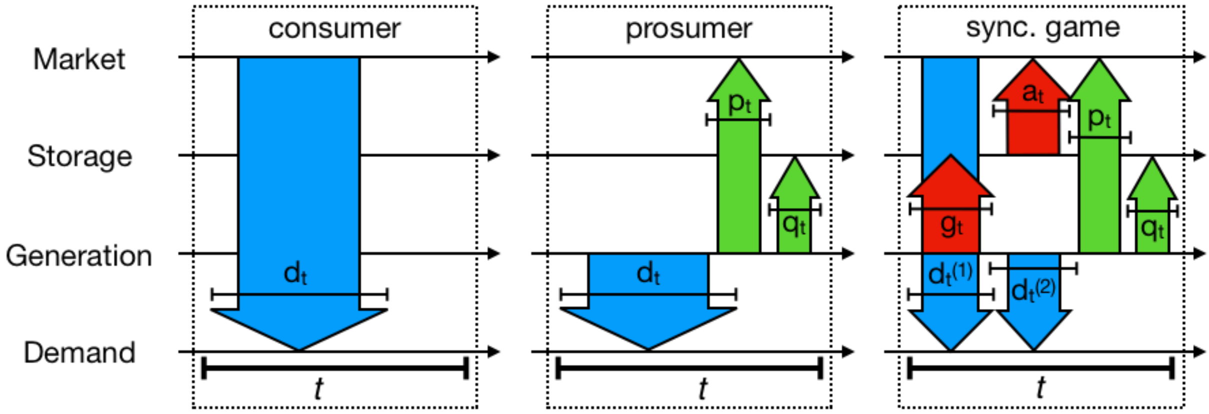


Figure 2.2: Example of how a synchronisation game works. The fist plot depicts a consumer, the middle plot a "friendly" prosumer showing normal prosumer behaviour and the third plot shows the actions of a prosumer who is playing a synchronisation game. The x-axis for all plots consists of one discrete time step t, in which the consumer and the prosumers execute actions. A consumer can only perform one action: buying energy from the market in order to fulfil his demand. When a prosumer decides to behave normally, he generates energy to fulfil his demand and sells surplus energy on the market (p_t) or stores it into a local storage (q_t). When a prosumer is playing a synchronisation game he splits his demand into two parts: $d_t^{(1)}$ and $d_t^{(2)}$. He would be able to fulfil the demand $d_t^{(1)}$, but instead he buys the same amount of energy from the market, artificially increasing the demand on the market. At the same time the prosumer is generating energy g_t and puts it into his local storage. In the subsequent step the prosumer generates the demand $d_t^{(2)}$ and sells the energy a_t from storage, which includes the previously generated energy g_t. For the prosumer this is a zero-sum game, although a small loss can be tolerated, since it will be amortised due to the prosumer being able to sell his generated energy for higher prices in the future. But in order to stay profitable, the prosumer has to minimise this loss.

ated, since due to the increased price they will be paid off when the prosumer can increase the price of his energy s_t^p. But the losses coming from a synchronisation game have to be kept at a minimum for the prosumer to maximise his profits from playing a synchronisation game.

2.1.5 Repeated Synchronisation Games

Playing only one synchronisation game may not result in an increased price from the network operator and thus might not be worth the effort, so prosumers have to play synchronisation games for multiple rounds. If a synchronisation game was successful and the network operator increases the price s_t by Δ_s, the prosumers may follow by increasing their prices s_{t+1}^p. But at the same time they have to consider that then there is plenty of cheap energy available on the market and therefore they will not be able to sell their energy for the higher price. To make sure that their generated energy can be sold on the market, a better strategy for the prosumers is to adhere to their old price s_t^p, but decrease the amount of energy $d_t^{(1)}$ bought from the market, so now they have to minimise the following function:

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s) - (a_t + \Delta_a) \cdot s_t^p \longrightarrow \min \tag{2.2}$$

The minimum can be replaced with 0, further we assume that the rates of consuming and storing energy are the same: $(a_t + \Delta_a) = d^{(1)} + \Delta_d$. Now we add $+\Delta_d - \Delta_d$ to this term: $d^{(1)} + \Delta_d + \Delta_d - \Delta_d = d_t^{(1)} + 2\Delta_d - \Delta_d$, keeping it equivalent, but simplifying the subsequent steps:

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s) - (d_t^{(1)} + 2\Delta_d - \Delta_d) \cdot s_t^p = 0 \tag{2.3}$$

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s) - (d_t^{(1)} - \Delta_d) \cdot s_t^p - 2\Delta_d s_t^p = 0 \tag{2.4}$$

Now we apply the same trick to the term $-(d_t^{(1)} - \Delta_d) \cdot s_t^p$, keeping it equivalent, but simplifying the subsequent steps: $-(d_t^{(1)} - \Delta_d) \cdot s_t^p + (d_t^{(1)} - \Delta_d) \cdot s_t^p - (d_t^{(1)} - \Delta_d) \cdot s_t^p = -2s_t^p(d_t^{(1)} - \Delta_d) + s_t^p(d_t^{(1)} - \Delta_d)$

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s) - 2s_t^p(d_t^{(1)} - \Delta_d) + s_t^p(d_t^{(1)} - \Delta_d) - 2\Delta_d s_t^p = 0 \tag{2.5}$$

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s) - 2s_t^p(d_t^{(1)} - \Delta_d) + s_t^p(d_t^{(1)} - \Delta_d) = 2\Delta_d s_t^p \tag{2.6}$$

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s) + s_t^p(d_t^{(1)} - \Delta_d) = 2\Delta_d s_t^p + 2s_t^p(d_t^{(1)} - \Delta_d) \tag{2.7}$$

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s) + s_t^p(d_t^{(1)} - \Delta_d) = \cancel{2\Delta_d s_t^p} + 2s_t^p d_t^{(1)} - \cancel{2\Delta_d s_t^p} \tag{2.8}$$

$$(d_t^{(1)} - \Delta_d) \cdot (s_t + \Delta_s + s_t^p) = 2s_t^p d_t^{(1)} \tag{2.9}$$

$$d_t^{(1)} - \Delta_d = \frac{2s_t^p d_t^{(1)}}{s_t + \Delta_s + s_t^p} \tag{2.10}$$

Equations 2.5 to 2.10 show how we end up with relationship between the reduction in purchased energy and the increase in price. Since the right side of equation 2.10 always has to be a minimum we can drop the factor 2 and since the prices s_t and s_t^p are fixed we can sum them up in a constant x, so the final equation is:

$$d_t^{(1)} - \Delta_d = d_t^{(1)} \cdot \frac{x}{x + \Delta_s} \tag{2.11}$$

Or with respect to Δ_d:

$$\Delta_d = d_t^{(1)} \cdot \frac{x}{x + \Delta_s} + d_t^{(1)} \tag{2.12}$$

As equation 2.12 suggests, the higher the price difference Δ_s rises, the less energy the prosumer can consume and still be profitable. The relationship between the increase in price and the amount of energy a prosumer can buy from the market is reciprocal, so even for very large increases in price there will always be a certain amount of energy (> 0) the prosumer can buy from the market and still be profitable while increasing the demand, given that the assumption about the rates of consuming and storing the energy holds.

2.2 Related work

As mentioned earlier, the work presented in this chapter is based on a paper I co-authored with my supervisor Olga Saukh [75], this paper was submitted and accepted at the IEEE International Conference on Communications, Control and Computing Technologies for Smart Grids. Everything presented in this chapter follows this paper, but here it is further elaborated and I added more details, which we had to omit for the publication.

A relatively large body of work used game-theoretical approaches and analysis for the energy market [28] [87] [44] [96], trying to identify optimal strategies for the actors on the market. In general, energy trading is often modelled as a non-cooperative game, which is played in multiple rounds. Mechanism design has not gained a lot of traction in the game-theoretical study of the energy market yet [67].

There is also a fairly large body of work studying and analysing the energy market and the power grid in general, using agent based modelling [73] [55] [43] [3]. Agent based modelling was used to study the wholesale market as well as studying microgrids.

Timing plays a very important role in managing the power grid, a lot of research was done exploring methods to keep the overall energy network stable, by solving technical issues in power grids regarding the integration of renewable energy sources and improving power quality [35] [98]. To the best of my knowledge there is now prior work dealing with synchronisation behaviour of prosumers.

2.3 Model and Simulation

I used agent based modelling[11] to model the behaviour of consumers and prosumers on a microgrid. The microgrid $\mathcal{M}$ is modelled consisting of $\mathcal{N}$ agents, which can either be a *consumer* $c \in C \subseteq \mathcal{N}$ or a *prosumer* $p \in P \subseteq \mathcal{N}$. The prosumers use Q-learning to maximise their profits[12]. There is also a network operator connected to the grid, whose responsibility is it to balance the network and adjust its price s_t accordingly. In order to keep things simple, each agent can perform exactly one action during a discrete time interval t_n. There is also no change in supply or demand of energy on the market during one time step. The network operator sets the price s_t during the beginning of each time step t_n, this price includes the cost of energy generation, the cost of network usage and the maintenance costs of the network.

2.3.1 Agent Based Modelling

An agent based model is a computational model for simulating actions and interactions of agents. It is widely used in different scientific fields to model interactions of agents with each other and their environment. The key notion behind agent based models is that agents who behave based on a rather simple set of rules can show rather complex behaviour, being a computational model it allows the thorough study of those complex behaviours. Wilensky and Rand define an agent as a computational individual with particular properties and actions and agent based modelling as a form of computational modelling where a phenomenon is modelled in terms of agents and their actions [94].

For the purpose of this work, agent based modelling is used to create a simplified model of

[11] For more details on agent based modelling see section 2.3.1.

[12] For more details on Q-learning see section 2.3.2.

a microgrid, which only consists of a manageable amount of agents involved, who are trading electrical energy amongst each other. The number of actions they can perform is fairly limited[13], but the whole model is still expressive enough to show the dynamics that arise in microgrids, allowing to further analyse them.

2.3.2 Q-learning

Q-learning is a model-free reinforcement learning algorithm [91]. Reinforcement learning denotes a class of machine learning techniques where an agent receives rewards or punishments for performing actions in an environment with the goal of maximising the rewards for all following steps. Model-free means that the agent does not have a model of the world it is acting in. The environment is defined by a set of states S, the available actions are defined by a set of states A. The Q-values are usually stored in a matrix, which has the dimensions $|S| \times |A|$ and are indicators for the "quality" of a certain action in a given state. The Q-value matrix gets updated every step, according to the following formula:

$$Q(s_{t+1}, a_{t+1}) = (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot (r_t + \gamma \cdot max\, Q(s_{t+1}, a)) \tag{2.13}$$

Where α is the learning rate, $Q(s_t, a_t)$ is the Q-value for the current state, r_t is the reward received for performing the selected action in the current state and γ is the discount factor, a number between 0 and 1 that defines whether the agent values immediate rewards or potential future rewards higher.

Even though Q-learning was first introduced in the late 1980s [90], it is still relevant today and gained a lot of attention in recent years, due to new developments like Deep Q-learning, which enabled an artificial agent to achieve super-human performance on some classic Atari video games, only being trained on the raw pixel output of the games, just like a human would perceive a game [59].

Since the actions and states of the problem domain of this particular problem domain are rather limited, I did not use Deep Q-learning, just standard Q-learning. Q-learning in this model is used to show that prosumers, whose only goal is it to maximise their profits are able to learn that playing synchronisation games is the more profitable than behaving "friendly", therefore, it is the preferred strategy during certain times of the day.

[13] As further elaborated in section 2.4.

2.4 Implementation Details

The simulation was implemented using Python and Numpy, the Q-learning algorithm was taken from an implementation by Steven Brown [12], which itself is taken from Mofan Zhou [101]. It consists of $|C| = 20$ consumers, $|P| = 5$ prosumers and one network operator. The time interval for one time step t_n is set to 15 minutes. Each agent can perform one action during this time step. A consumer can only consume energy, a prosumer has two actions to chose from:

1. **Be "friendly":** The prosumer generates energy for its own needs and sells the surplus on the market. If it is able to sell some energy, it increases the price for the remaining energy by 10%. If the current price is too high (higher than the price set by the network operator), it decreases the price by 10%.

2. **Play a synchronisation game:** As explained in section 2.1.4 the prosumer neither gains nor loses money (although a small loss can be tolerated), but it artificially increases the demand on the market, leading to a higher price set by the network operator.

The Q-learning algorithm of the prosumer decides which action to use according to the current *tick*, which denotes the current time of the day. Since one tick indicates that 15 minutes have passed, there are $\frac{60}{15} \cdot 24 = 96$ ticks during one day[14]. During the training phase of the algorithm, which lasts for 50000 ticks, which corresponds to 520 days and 5 hours, the Q-values for each prosumer get updated during each single time step. Once this is one, the algorithm runs for 500 more ticks which corresponds to 5 days, 1 hour and 15 minutes and without updating its Q-values.

One step in the simulation is done by first letting all consumers and prosumers chose their actions. The only action a consumer can take is to consume energy, therefore the action of a consumer is to determine its current energy demand. A prosumer can decide between the above described actions, its current demand depends on which action it chose. Once the energy demand for all agents is determined, the agents can buy and sell energy on the market. The order in which the agents buy energy from the market is randomised and therefore always different. The agents buy energy from the market in order to fulfil their demands, where they always buy the cheapest energy first. If a prosumer sells some of its generated energy, it increases its energy price by 10%, if the energy price of a prosumer at the end of one time step t_n is higher than the price from the network operator, the price gets decreased by 10%. The initial energy price of a prosumer is the baseline price of the network operator.

[14] A tick is one time step t_n, but the counter for ticks get reset at the beginning of each day

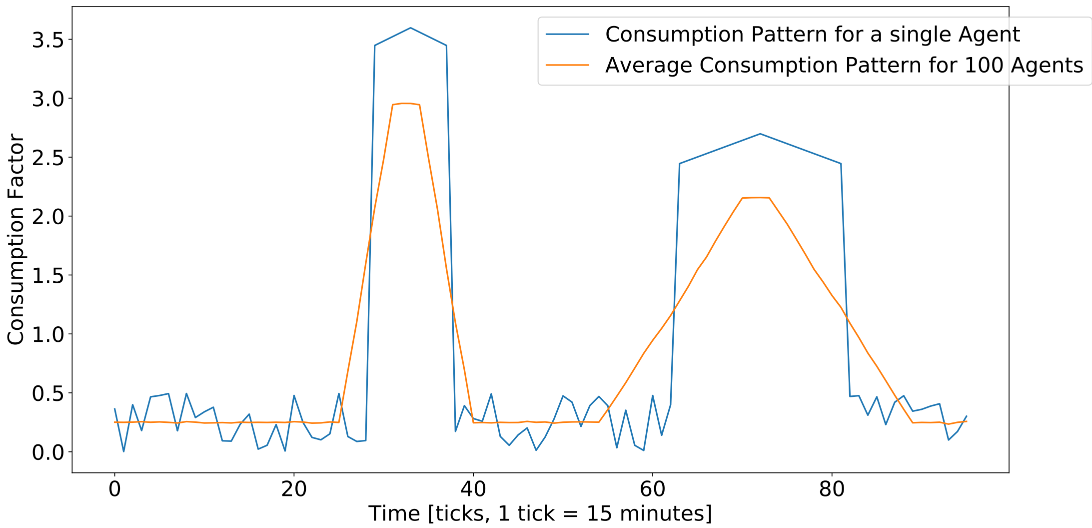


Figure 2.3: The consumption pattern for a single agent and the averaged consumption patterns for 100 agents for one day (96 ticks). Compared with the averaged values, the agent in this figure has higher consumption peaks in the morning and in the afternoon. The consumption is also rather noisy when not approaching a peak, the peaks appear rather saltatory. The average consumption on the other hand is much smoother, especially when approaching the peaks.

After all agents performed an action and bought energy on the market[15], the network operator checks during each time step, whether each agent consumed more energy than its baseline demand. If this is the case, this agent has to pay the additionally consumed energy volume to the network operator and the excess energy gets added to the total energy balance. If this total energy volume is higher than the baseline volume, the network operator adjusts its current price accordingly:

$$s_{t+1} = s_t \cdot \left(\frac{\text{total\_energy\_volume}}{\text{baseline\_volume}} \right)^2 \tag{2.14}$$

This new price gets reset to the base price at the beginning of each day, as shown in figure 2.7.

Computation of Consumption and Generation

Figures 2.3 and 2.5 show the consumption/generation pattern for a single agent/prosumer as well as an averaged pattern for 100 agents/10 prosumers. Especially the averaged patterns show how the consumption/generation varies throughout one day. The consumption has two peaks,

[15] Buying or selling energy on the market does not count as one action, since it happens automatically on the microgrid.

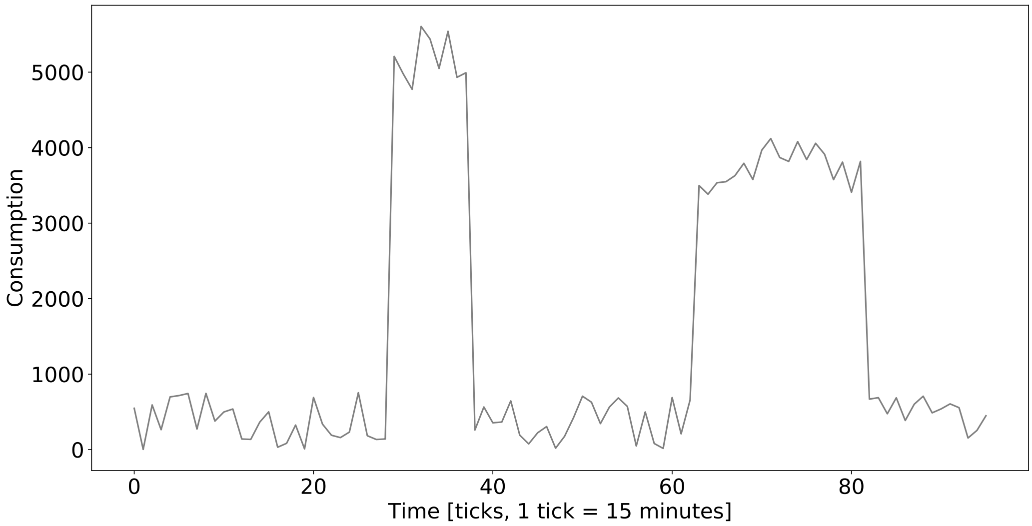


Figure 2.4: The actual consumption curve for a single agent. This consumption curve is taken from the agent, whose consumption pattern is depicted in figure 2.3. Both figures show the course of the consumption pattern or consumption curve over one day (96 ticks). The actual consumption roughly follows the consumption pattern, especially with respect to the peaks. To calculate the actual current consumption, the current consumption factor for the current time step is taken from the consumption pattern and multiplied by the consumption per time step. The resulting value is used as centre of a normal distribution from which the actual consumption value will be sampled.

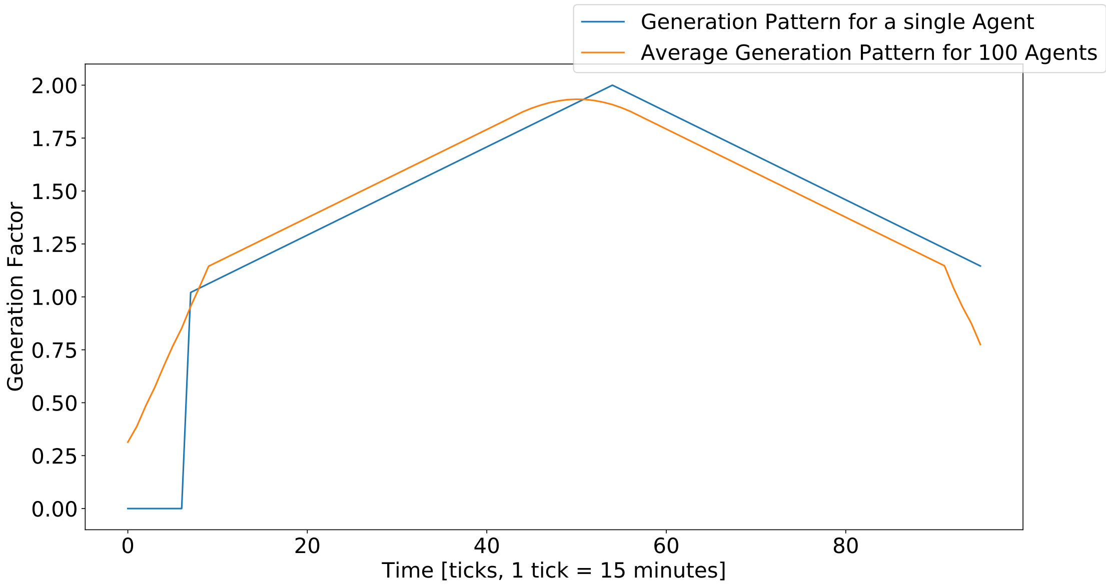


Figure 2.5: The generation pattern for a single prosumer and the averaged generation pattern for 10 prosumers for one day (96 ticks). The peak generation of this prosumer is slightly after noon and the average generation never reaches zero. These abstractions were done for our model to create a permanent supply of producer generated energy, which can be traded on the market. The longer time windows, where energy can be sold also works as an abstraction for the prosumers' local energy storages.

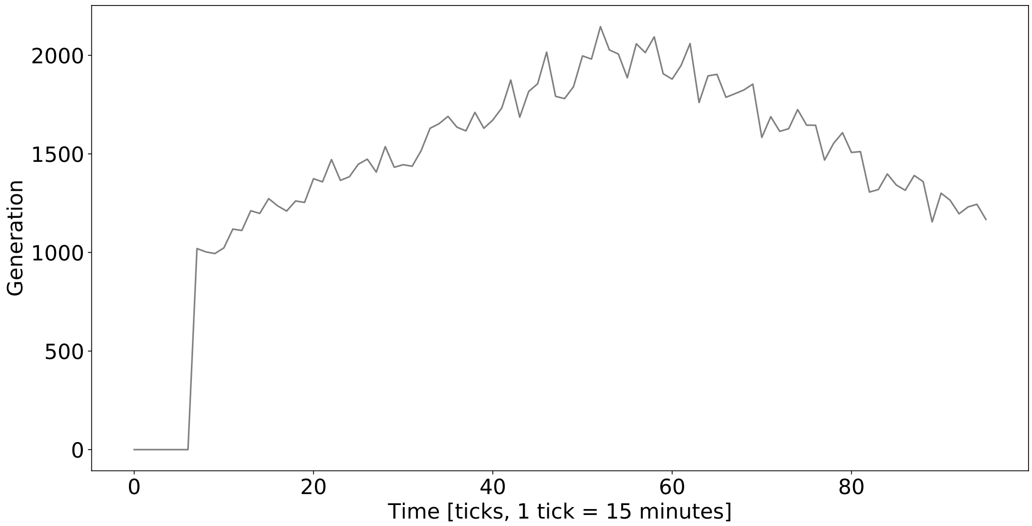


Figure 2.6: The actual generation curve of a prosumer for one day (96 ticks). This is the amount of energy the prosumer from figure 2.5 generates during one day. The relationship between this figure and figure 2.5 is the same as between figure 2.3 and figure 2.4: the actual generation gets sampled from a normal distribution, where the centre of the normal distribution is set by the generation factor for the current time step multiplied by the generation per time step. Compared to figure 2.4, the variance for the generation is much lower.

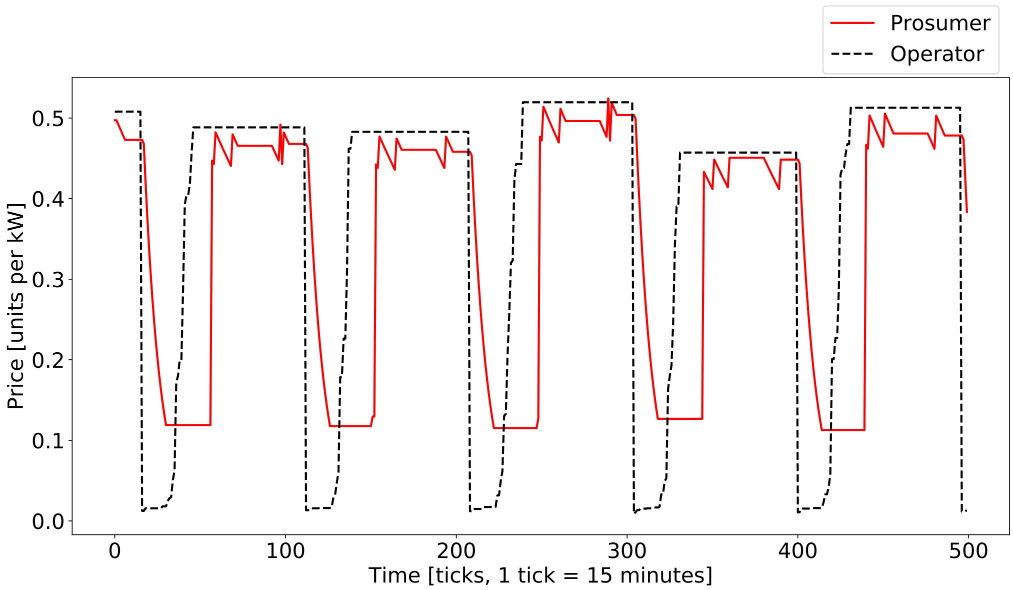


Figure 2.7: The change of the network operator's price and one prosumer's price for 5 days, 1 hour and 15 minutes (500 ticks). This price curve was taken after the training of the Q-learning algorithm was done and the prosumers learned to synchronise their behaviour. The network operator's price gets reset to zero at the beginning of every day (around tick 100), and steeply increases around the time of the morning peak for each day. The prosumer follows this price, when the network operator's price is low, the prosumer is slowly converging towards this price, but as soon as the network operator increases its price, the prosumer also increases its price. The prosumer in this plot tended to slightly oscillate around the network operator's price with its own price, in the first and third spike it even slightly overshot the price of the network operator. Note that the plot starts during the day and not at the beginning of a day.

one in the morning and one in the afternoon. The generation has only one peak, usually around noon. Figures 2.4 and 2.6 show two actual examples for consumption and generation. The aforementioned peaks are also visible here, but those two plots also show that there is still some randomness in the consumption or generation. This randomness is required in order to better simulate real consumption or generation. To calculate the actual consumption or generation, the consumption or generation per tick is multiplied with the consumption or generation factor taken from the consumption or generation pattern for the current time step. This value is then used as the centre of a normal distribution from which the actual value for the current time step is sampled from:

$$\texttt{consumption\_annual} \sim \mathcal{N}(\texttt{ANNUAL\_CONSUMPTION\_CAPITA} \cdot n_{\texttt{persons}}, \sigma_{acc}) \tag{2.15}$$

$$\texttt{consumption}_t = \frac{\texttt{consumption\_annual}}{365 \cdot \texttt{TICKS\_DAY}} \cdot \texttt{consumption\_factor}_t \tag{2.16}$$

$$\texttt{actual\_consumption} \sim \mathcal{N}(\texttt{consumption}_t, \sigma_c) \tag{2.17}$$

Equation 2.15 denotes how much an agent consumes per year, this value is also sampled from a normal distribution. `ANNUAL_CONSUMPTION_CAPITA` is a constant set to $16.9 \cdot 10^6$, this value was taken from the annual consumption per capita from New Brunswick, Canada [15], which is multiplied with number of persons living in the household of an agent. The variance σ_{acc} is set to be 15% of `ANNUAL_CONSUMPTION_CAPITA`. Equation 2.16 denotes how the consumption per time step is calculated from this variable, `TICKS_DAY` is a constant set to the number of discrete time steps per day, since in this simulation the duration of a time step is set to 15 minutes, $\texttt{TICKS\_DAY} = \texttt{96}$, the consumption per time step can be calculated by dividing the total consumption per year by the number of time steps per year. The $\texttt{consumption\_factor}_t$ is the current consumption factor, taken from the consumption pattern. For the actual consumption per time step this previously calculated value is used as mean value (centre) of a normal distribution, from which it will be sampled. The variance σ_c is again set to be 15%, but this time from $\texttt{consumption}_t$.

The values for the actual generation per tick are calculated similarly:

$$\{\texttt{generation\_panel} \in \mathbb{Z} | \texttt{generation\_panel} \in \{235, \dots, 435\}\} \tag{2.18}$$

$$\texttt{generation}_t = n_{\texttt{panel}} \cdot \texttt{generation\_panel} \cdot \frac{\texttt{MINUTES\_TICK}}{60} \cdot \texttt{generation\_factor}_t \tag{2.19}$$

$$\texttt{actual\_generation} \sim \mathcal{N}(\texttt{generation}_t, \sigma_g) \tag{2.20}$$

Equation 2.18 defines how much energy a single panel of a prosumer can generate. The exact amount is an integer between 235 and 435 watt hours, the values are inspired from generation of a typical solar panel which can be bought today [2], the maximum generated value is set slightly higher than what today's top solar panels can generate, so that the prosumers can compete on the market. The generation per time step is denoted in equation 2.19, it is simply the number of installed solar panels[16], multiplied by the generation of one panel[17], multiplied by the number of time steps per hour[18], multiplied with the current generation factor, which is taken from the generation pattern. This value is then used as the mean value (centre) of a normal distribution, from which the actual generation is sampled, as denoted in 2.20. The variance σ_g is set to be 5% of this value, which can be seen in figure 2.6, where the plot is much less noisy than in figure 2.4.

2.5 Evaluation of the Results

Figure 2.8 depicts the results of the simulation after the training phase. Plot 1 shows that prosumers are able to learn that driving up the demand peak hours is a profitable strategy, especially prosumer 1, whose energy demand has especially high peaks during the peak hours and also after the peak hours are over. Plot 2 contrasts the energy that could potentially be generated with the energy that was actually generated for two prosumers. This is a good indicator for how much influence the synchronisation games have on the network. Plot 5 shows when prosumers play a synchronisation game: 3 out of 5 prosumers in the microgrid learn to synchronise their actions during both peaks. It is especially remarkable that the prosumers learned to play a synchronisation game for the afternoon peak happens at exactly the same time. This plot follows the consumption curve to a certain degree: during the peak in the morning and the peak in the afternoon it is most profitable for a prosumer to play a synchronisation game. An example of how much profit two prosumer and one consumer make is shown in plot 4. Shortly after the prosumers played synchronisation games during the morning peak, their profits peak as well, the consumer on the other hand just loses money during one day, because it has to buy extra energy from the market. Plot 3 is the same plot as shown in figure 2.7, but

[16] Which is also a randomly sampled integer between 10 and 25.

[17] This model assumes that a prosumer buys multiple panels of they same type, so all of them have the same generation capacity.

[18] `MINUTES_TICK` is a constant denoting how many minutes pass with each time step, it is 15 minutes in this simulation.

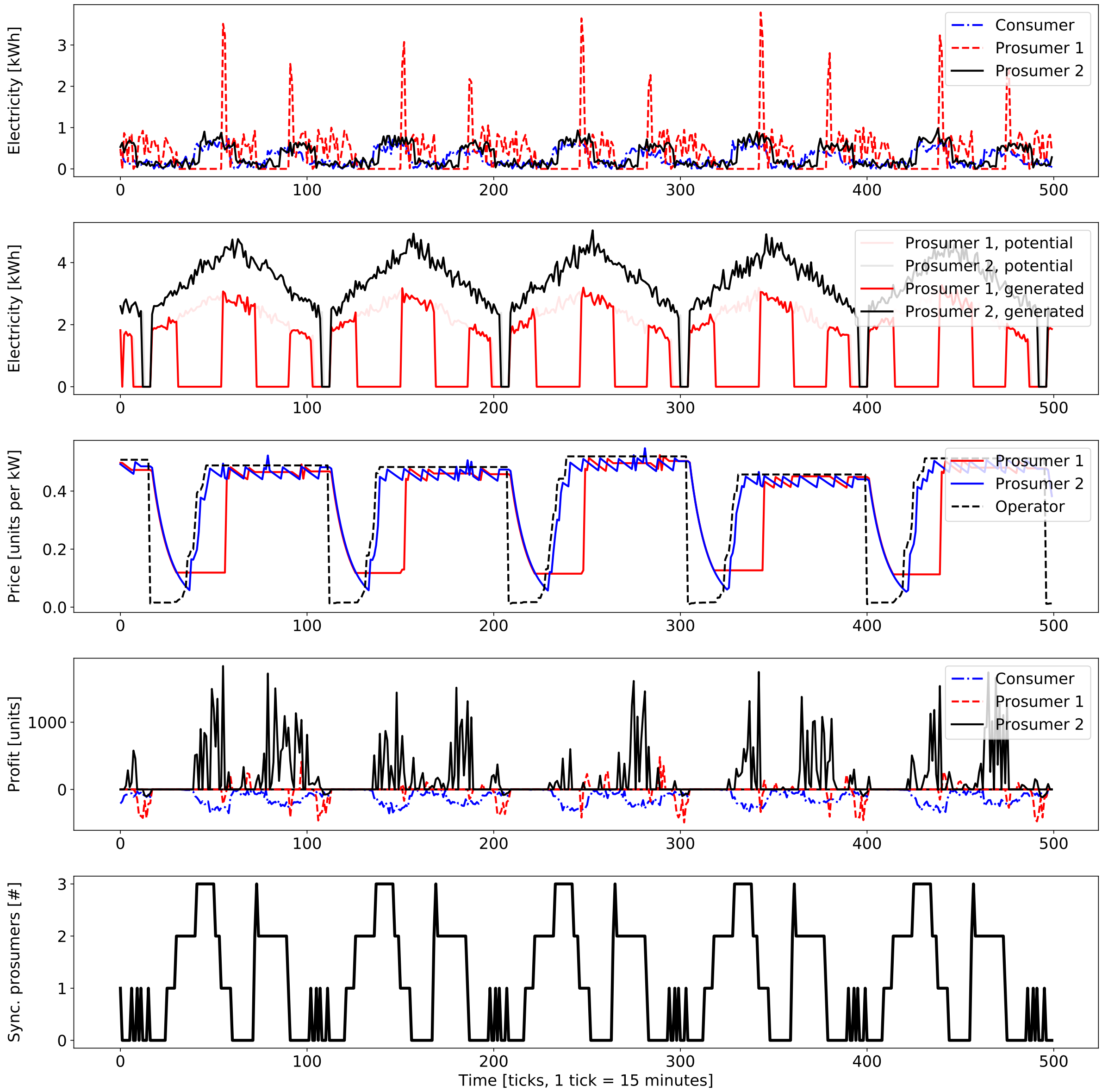


Figure 2.8: The results of the simulation, after the prosumers' Q-learning algorithms are trained. Those five plots show the following:

1.) The consumption curve for one consumer and two prosumers.

2.) The generation curve for two prosumers, for the potential and actually generated energy.

3.) Prices for two prosumers and the network operator.

4.) Profits made by one consumer and two prosumers.

5.) Number of prosumers playing a synchronisation game.

These plots all span 5 days, 1 hour and 15 minutes or 500 ticks, after the prosumers learn to play synchronisation games.

it features two prosumers and sets their prices in relation to the other plots. Like in figure 2.7 all the plots in figure 2.8 are slightly shifted and do not start at the beginning of a day.

2.6 Conclusion

As section 2.5 shows (selfish) prosumers are able to learn how to synchronise their actions in order to make higher profits. This behaviour comes with many negative drawbacks for the other parties involved on a microgrid, as well as for the environment: consumers have to pay higher prices for electrical energy than they would pay otherwise and for the network operator the artificially created demand means a higher load on the network. In order to compensate for this higher load, the spinning reserve has to be activated, which is expensive, wasteful and also has a negative impact on the environment, since it heavily relies on burning fossil fuels or gas. From a prosumer's perspective this behaviour corresponds to a Nash-Equilibrium [60], but a Nash-Equilibrium that fails to maximise social welfare, in fact it rather decreases social welfare because it leads to higher consumer prices and has a negative environmental impact.

2.6.1 What makes Synchronisation Games possible?

Synchronisation games are possible, because prosumers have much more influence on what is happening on the power grid than regular consumers do. Consumers have in general only very limited influence on the load on the network, e.g. by joining a demand side response programme, and no interest in artificially raising it, since it comes with higher prices for them. Prosumers on the other hand side have the interest to raise prices and also the means, especially when there are a lot of prosumers on a microgrid. Such a synchronisation game can be realised with relative ease in practice: the prosumers just need to install a software that claims to increase their revenues from selling energy. The software uses standard time synchronisation protocols like NTP [57] to determine when a synchronisation game should be played. A prosumer can be further incentivised to get other prosumers to install such a software, by a referral programme, set up by the creator of this software. The more prosumers use such a software, the more effective it will be. It is also worth noting that research shows that people in general are more likely to collaborate than rational mind would do [32].

2.6.2 How can Synchronisation Games be detected and prevented?

Smart meters [61] play a key role in detecting synchronisation games. However, they do have some shortcomings like a low temporal resolution and not being able to deliver measurements in real time[19]. But even with these drawbacks, smart meters can be used by the network operator together with machine learning methods for anomaly detection to discover unusual high loads, which could indicate that synchronisation games are being played.

Improvements in the accuracy of smart meters together with real time reporting could help to avoid manipulations, as described in section 2.1.4. The smaller the time interval in which prosumers can play synchronisation games, the less profitable it is for them to do so, also real time pricing will lead to a reduced peak demand, which decreases the profitability of synchronisation games. Similarly, also proper incentive mechanisms designs, like e.g. receiving decent monetary compensation for participating in a demand-side response programme [29] could make playing synchronisation games less attractive. Finally, making the price s_t more resilient to fluctuations of the daily load, i.e. by having a spinning reserve that is cheaper to activate and operate, could also decrease the profitability of synchronisation games and therefore the incentive to play them.

[19]Usually smart meters only provide a measurement every 15 minutes [4].

Chapter 3

Detecting Solar Panels from High Resolution Aerial Images

3.1 Problem Definition

Solar power or photovoltaic (PV) systems are on the rise. Due to falling prices for photovoltaic arrays [78], installing a solar panel to generate electrical energy for customer's own consumption or for selling it on a microgrid[1] is becoming a viable option for more and more people, especially homeowners. Because of falling acquisition costs for solar panels and the savings in electricity costs from the energy they generate, solar panels gained more and more widespread adoption. Installing a solar panel and setting up a PV system can nowadays be done relatively easily. Because of this, solar panels are more and more found on the roofs of private homes or commercial structures.

To better understand what the current state of solar panel adoption is and at which rates their adaption grows, in order to better analyse and evaluate the driving factors behind it, it is interesting for government agencies, utility companies and third parties to gather information about the current state of PV system adoption, like location, power capacity and energy production. Several organisations have begun to collect and publish such information, this is usually done using surveys [52] or projections [9] and therefore cumbersome, costly and limited in their accuracy. Thus, for many regions there exists no comprehensive data about the spreading and adaption of solar panels. Furthermore, states, districts and municipalities can handle the census of existing PV systems as they please, resulting in this data either being non-existent or spread over different data bases[2]. Since the installation of solar panels is usually heavily subsidised,

[1]Microgrids are further explained in section 2.1.3.

[2]The legal frameworks for installing a solar PV system are quite heterogeneous. In South Africa for example

looking at data bases containing this data could be an option to get better insights about the current state of PV systems for a certain area. But also in this case this data is not always public and only contains information about PV systems that got subsidised, which can vastly differ from the actual number of installed photovoltaic systems. As figure 3.1 shows, in Austria for example as of November 2018 only 6 out of the 9 federation states are subsidising the installation of PV systems and there also exist a subsidy for all of Austria, which shows how fragmented the subsidies are. At the time of writing, the only available data sets for Austria regarding PV systems were from installed PV systems in Lower Austria [62] and subsidised PV systems in Vienna [92]. For Europe and the rest of the world the situation is similar: data sets for solar PVs are i.e. provided for the city of Leeds [20] or Sydney and bordering parts in New South Wales [68]. Those data sets have a slightly different focus, since they also contain collected meter readings in kWh. In order to protect the privacy of the residents, the location data is limited to the postcode for domestic systems. A similar approach is used by the Department of Energy in the United States for the *Open PV Project* [26], unlike the previous mentioned data sets, it tries to compile a comprehensive database of PV installation data for the whole United States.

In this work I am going to propose a system that is able to recognise solar panels from high resolution aerial images to enable an automatic census of installed photovoltaic systems for a given area.

3.2 Related Work

In 2016 a data set consisting of high resolution aerial images for photovoltaic array detection and remote sensing object detection was released [11]. The release of this data set was accompanied by a paper, which uses a random forest classifier to set a baseline for future performance measurements [52]. Following these publications, the authors also published works using this data set for training again a random forest classifier [53] or a deep convolutional neural network [50] as well as studying other aspects of this data set [51] [39] [14] [89]. Other authors also use convolutional neural networks [58] or more traditional approaches like line primitive association analysis with template matching [88] or adaptive clustering [99]. Object detection to detect

residents of Cape Town have to register their solar panels by February 2019 [21], while residents of other parts of South Africa do not have to register yet. City officials defended this decision, claiming that it would only be a matter of time until registering solar PV systems becomes mandatory by the Energy Regulator of South Africa (NERSA) [22]. In Germany for example this is already the case [13], while in Austria the criteria for mandatory registration is set by the federal states. E.g. in Lower Austria solar PV systems with more than 50 kW have to be registered [23, §15], while in Styria solar panels have to be registered, if the area of their collector surface is less than 100 m^2 and they are less than 3.50 meters in height [24, §21].

Figure 3.1: Subsidies for PV systems in Austria. Federation states with solar panel icons show which states subsidise the installation of energy storage systems, which is done by seven out of nine states, the federation states with battery icons show which states subsidise the installation of PV systems, which is done by six out of nine states. The request for proposals for the state of Styria was closed on 16th May 2018, the state of Vorarlberg is not subsidising as of November 2018, due to the lack of public funds. This figure is taken from [6].

solar panels from aerial images is a rather new field of research, but it was already applied in the detection of roads [10] [83] [7], buildings [97] [16] [33] or vehicles [49] [17] [41].

3.2.1 Image Recognition vs. Object Detection

Image recognition is to identify what is depicted in a given image, object detection is to identify what objects are present in a given image and to give their location. Image recognition has a long history in computer vision and machine learning [31], e.g. for fields like face recognition [85]. In order to train such a system, one needs a set of labelled data that is fed into a supervised machine learning algorithm and the algorithm tries to minimise the prediction error on the training data set. Systems for object detection are usually trained similarly: labelled data is fed into a supervised machine learning algorithm and the error on the training data set should be minimised. Creating a training data set for object detection requires more effort, since the annotation process needs more work, because the location of objects in the image as well as their labels have to be annotated, therefore the amount object detection datasets are limited, compared to datasets for image detection [70].

Breakthroughs in image recognition, where using convolutional neural networks outperformed all competing approaches in the image net challenge [74] by a magnitude on a standardised dataset. This started the current trend of deep learning back in 2012. Advances in object detection took a bit longer, but in recent year also the performance of object detection algorithms has increased tremendously. New approaches like *You Only Look Once* have a reasonably good detection performance and are also fast enough to perform object detection on videos almost in real time [72].

3.2.2 You Only Look Once (YOLO)

The algorithm I adopt in this work for object detection is called *You Only Look Once* (YOLO), it was first introduced in 2016 [72], improved versions were released later that year, called “YOLO9000: Better, Faster, Stronger” [70] and in 2018 called “YOLOv3” [71]. As the name suggests, YOLO performs object detection by only evaluating a picture once, unlike other approaches like sliding windows, which iterate over an image for multiple times. This allows for object detection to be performed much faster compared to other state-of-the-art approaches in object detection.

YOLO is formulating object detection as a regression problem instead of a classification problem. Unlike sliding window and region proposal-based approaches, YOLO sees the whole image during the training and test phase, so it gets the full context of an image. The image is divided into an $S \times S$ grid, where each grid cell is responsible for detecting an object when its centre falls into one grid cell. Each grid cell predicts B bounding boxes with confidence scores for each bounding box, each bounding box consists of 5 predictions: x, y, w, h and the confidence score. The coordinates(x, y) represent the centre of a bounding box, w and h represent its width and height, which are predicted relative to the entire picture. The confidence score represents how well the algorithm assumes that the bounding box belongs to the stated class, multiplied by how accurate it assumes the intersection over union (IOU) between the predicted bounding box and the ground truth to be: $c = P(\text{Object}) \cdot \text{IOU}_{\text{pred}}^{\text{truth}}$. For the first version of YOLO, the algorithm used fully connected layers on top of the convolutional feature extractor to predict the coordinates of the bounding boxes, this meant that for each grid cell only one object class could be predicted[3], regardless of how many bounding boxes B were predicted for a certain grid cell. The authors used the PASCAL VOC data set [27] to test their algorithm and set the number of grid cells $S = 7$ and the number of bounding boxes $B = 2$, which led to the system being able to identify a total of $7 \cdot 7 \cdot 2 = 98$ objects in one image. For the second version of YOLO, they removed the fully connected layers and used anchor boxes instead, by doing so they were also able to decouple the class prediction from the prediction of the spatial location and instead predict classes for each anchor box. Using anchor boxes allows the algorithm to now be able to identify more than a thousand objects for one image using the same hyperparameters as previously used for testing the first version of the algorithm on the PASCAL VOC data set. The mean average precision (mAP) slightly decreased from 69.5 to 69.2 in YOLO version 2, but at the same time the recall increased from 81% to 88% in the second version of the algorithm.

Apart from introducing anchor boxes, the second version of YOLO also introduces additional improvements like *batch normalisation*, *high resolution classification*, *fine-grained features* or *multi scale training*. Introducing anchor boxes also introduced new issues the authors needed to handle. The first issue stems from the fact that anchor boxes needed to be hand-picked. The network is able to learn how to adjust the boxes appropriately, but selecting better priors makes it a lot easier for the network to learn good predictions. Instead of hand-picking priors, the authors ran k-means on the bounding boxes of the training set to automatically find good priors. Another issue when using anchor boxes is model instability, especially in early iterations of the algorithm. Usual approaches using hand-picked priors learn from minimising the offsets to the anchor boxes they predict, YOLO version 2 directly predicts the coordinates of the objects in a grid cell. This direct prediction is locally constrained and therefore more stable during the early iterations of the training. YOLO version 3 introduced some minor improvements like

[3]E.g. if a grid cell contained the centre of one cat and one dog, only one of them could be correctly identified, if a grid cell contained two centres of two cats, both could be correctly identified

using independent logistic classifiers instead of using softmax. The biggest change is that YOLO version 3 now predicts boxes on three different scales and extracting features from those different scales, similar to feature pyramid networks [46]. The system is trained to predict bounding boxes for three different scales, where the feature maps from the predictions for smaller scales are upsampled by a factor of two and merged together with outputs from further down the network, to get more meaningful semantic information from the upsampled features and finer grained information from the earlier features[4]. The final prediction benefits from incorporating the prior computations with the fine-grained features from early on in the network. All version of YOLO were implemented in C, using the darknet framework [69] and the code is open source.

Figure 3.2 shows how YOLO is able to detect two objects in one image, even though they have overlapping bounding boxes, detecting the objects in this picture took 2.72 seconds when starting the script to detect for the first time and less than one second when identifying afterwards on an Intel Core i7-7700HQ CPU. Figure 3.3 shows when YOLO is not able to correctly identify an object in a picture. It is able to correctly locate the object of interest, but since it was not trained on a data set containing images of bunnies, it is not able to correctly identify it as a bunny. The algorithm took again 2.72 seconds to identify the location of the bunny when launched for the first time and less than one second for subsequent images on an Intel Core i7-7700HQ CPU. When letting the algorithm run on a GPU, it would be faster by several magnitudes.

3.3 Architecture of the System

As show in figure 3.4, the input layer, shown in green consists of one ($m = 1$) convolutional neural network with 32 filters ($n = 32$) and batch normalisation and Leaky ReLU as activation function. All black boxes represent one block of hidden layers that consist of one input layer, a convolutional neural network with n filters, connected to m subblocks of two convolutional neural networks where the first convolutional neural network has $\frac{n}{2}$ filters and the second one has again n filters. The numbers next to the blocks indicate the number of filters and the number of layers with the according number of filters. E.g. the second block of hidden layers consists of 3 convolutional neural networks with $n = 128$ filters and 2 convolutional neural networks of $n = 64$ filters. The first layer with $n = 128$ is connected to one layer of $n = 64$ filters, which is connected to a layer of $n = 128$ filters, which is again connected to one layer of $n = 64$ filters and that layer is connected to the final layer of the block, which has again $n = 128$ filters. Each block consists of total of $2 \cdot m + 1$ layers. The kernel size of the convolutional neural networks with n filters, is (3,3), the kernel size for all convolutional neural networks with $\frac{n}{2}$ filters is (1,1).

[4]See figure 3.4 for a graphical representation of the system's architecture.

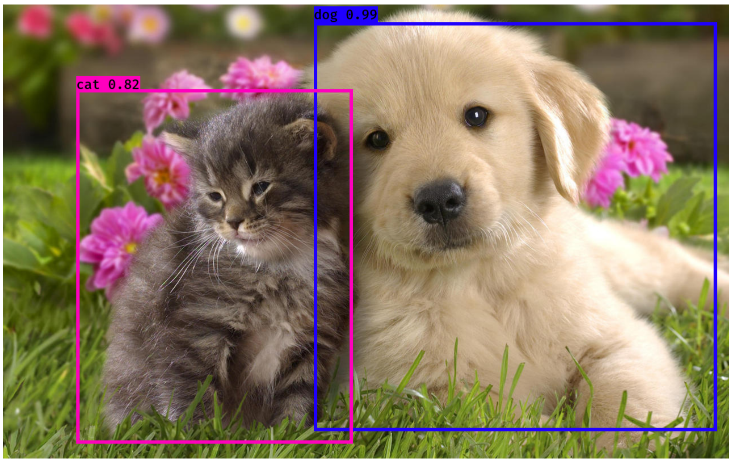


Figure 3.2: Example where YOLO is successfully identifying two objects in an image. The algorithm is succeeding even though the bounding boxes of the cat and the dog are overlapping. The dog is classified as dog with a confidence of 99%, while the cat is identified as cat only with a confidence of 83%. Image source unknown, found by Google.

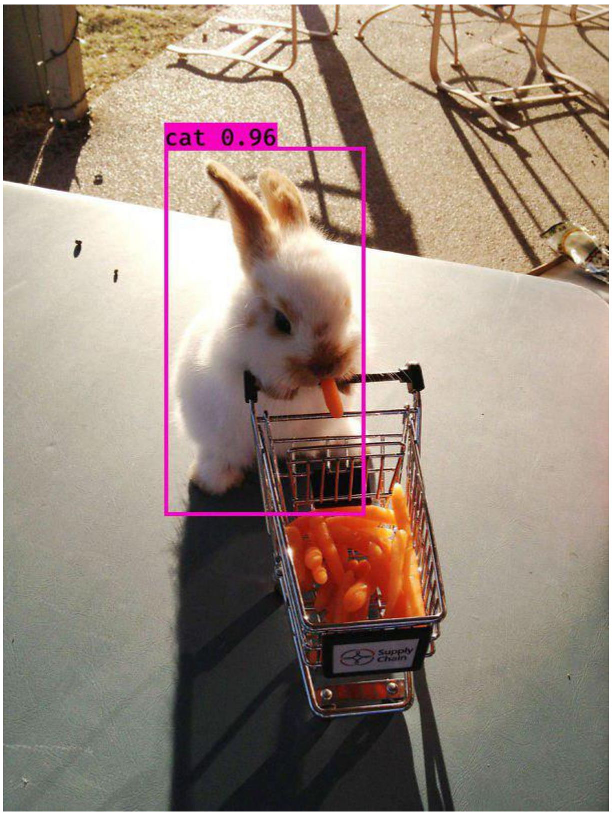


Figure 3.3: Example where YOLO fails to correctly identify the object in an image. The algorithm was able to predict the correct location of the bunny in the image, but since it was not trained on images of bunnies, only on images of cats, it identifies the bunny in the picture as a cat. The remarkable thing about this prediction error is, that YOLO's confidence in this prediction is 96%. Image source unknown, found by Google.

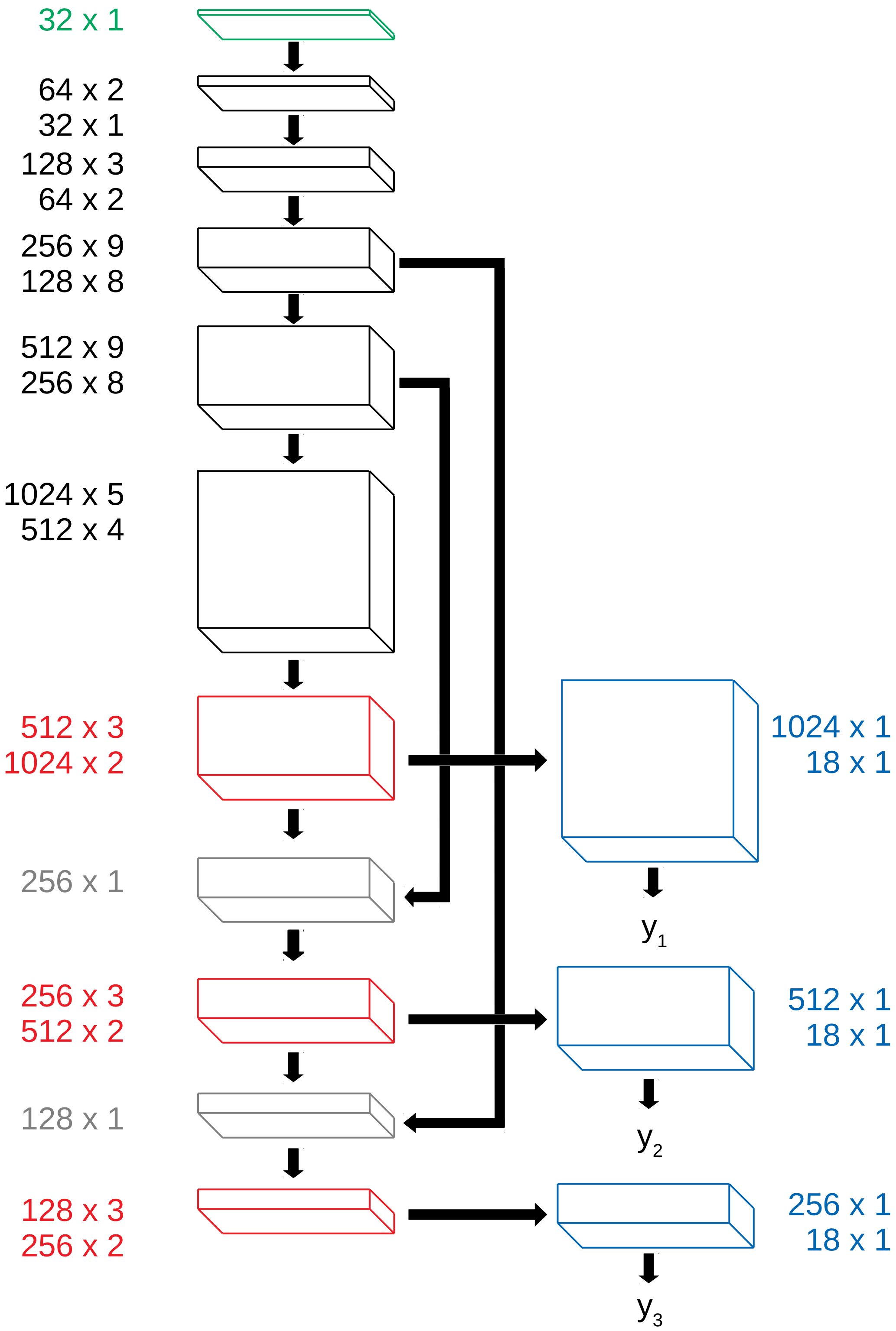


Figure 3.4: A schematic overview of the used neural network architecture. The box block represents a single convolutional neural network used as input, the black boxes represent blocks of convolutional neural networks making up the hidden layers, the red boxes represent blocks of convolutional neural networks making up the last layers and the blue boxes represent the output layers.

Stride sizes are in general (1,1), except for the first convolutional neural network of a block, which has a stride size of (2,2).

Boxes shown in red are the final blocks of the system and are connected to the output layers, shown in blue. The grey block between the two final blocks acts as an intermediate layer between the blocks for predicting boxes and predicting the confidence scores. The final block for the box prediction consists of a total of five convolutional neural network layers, three with $n = 512$ filters and two with $2n = 1024$ filters. Similarly, the final block for predicting the confidence scores consists of three convolutional neural network layers with a filter size of $n = 256$ and two layers with $2n = 512$ filters. Analogous to the blocks of hidden layers, a layer with n filters is connected to a layer of $2n$ filters, which is again connected to a layer of n filters, another layer of $2n$ filter and a final layer of n filters. Layers with n filters have a kernel size of (1,1), layers with $2n$ filters have a kernel size of (3,3). All neural networks in the final blocks and also the one for the intermediate layer are convolutional neural networks with batch normalisation and Leaky ReLU as activation function. The intermediate layer is just one convolutional layer, with 256 filters with and additional upsampling layer for 2D inputs at the end, which is responsible for doubling the input size coming from the output layer of a previous block the intermediate layer is concatenated with. The kernel size for the intermediate layer is (1,1). An intermediate layer is connected to the last block of a certain output and to the output of a previous block. The intermediate layer for y_2 is concatenated with the output from the fourth block, the intermediate layer for y_3 is concatenated with the output from the third block.

Output layers consist of two convolutional neural networks, they inherit their amount of filters from the final blocks they are connected to. The first convolutional layer of this block has $2n$ filters, while the final layer has again $n_{\text{out}} = 18$ filters, n_{out} is the number of output filters, which comes from the number of anchor boxes multiplied[5] by five predictions for each class:

$$n_{\text{out}} = n_{\text{anchors}} \cdot (n_{\text{classes}} + 5) \tag{3.1}$$

For this particular problem there is only one object class, we are interested in detecting: solar panels, therefore $n_{\text{classes}} = 1$, the number of anchors $n_{\text{anchors}} = 3$. The five predictions made for each class represent the upper and lower x and y coordinates of the bounding box: $x_{\text{min}}, y_{\text{min}}, x_{\text{max}}, y_{\text{max}}$ and the class this object belongs to. So, $n_{\text{out}} = 3 \cdot (1 + 5) = 18$. The sizes of the kernels are again (3,3) for the layer with $2n$ filters and (1,1) for the layer with n_{out} filters.

[5]Each output layer is responsible for performing the detection related to certain anchor boxes. The number of anchor boxes is set to $n_{\text{anchors}} = 3$, for one output layer, but the total number of anchor boxes is $n_{\text{anchors\_total}} = 9$, since there are three output layers.

3.4 Implementation Details

The proposed system implements YOLO version 3 in Python, using Keras [18] with a TensorFlow [1] back-end. My implementation is based on an implementation uploaded by an anonymous author to GitHub [5]. This implementation relies on pre-trained weights from the original Darknet implementation of YOLO [69].

In order to get YOLO to run properly, a proper workspace has to be set up. My implementation does this automatically when run for the first time[6]. The process of training the model is fairly standard: first the dataset is split into training and validation data, then the actual training starts with using the training examples for adjusting the weights of the neural network and the validation examples are used for testing the quality of the model. This is done until the maximum amount of epochs for training is reached, in this case it is after 50 epochs.

When the method for training the model is called, it first creates the model and adds additional features like logging, saving at checkpoints, reducing the learning rate on plateaus or early stopping to the model. The creation of the model is a rather complex process due to its structure[7]. The model is trained on images of three different input sizes for the same input image. Input images as well as their bounding boxes have to be resized accordingly. All input images for training as well as for prediction have to be resized according to the predefined size of the input layers of the convolutional neural networks which is (416×416). In order to keep the memory requirements low during training, the input images as well as the three true output values for those images are fed into the neural network using a generator, which does the scaling of the input image as well as the preprocessing for the bounding boxes on those scaled input images. The batch size for training is set to three, allowing the model to be trained using a notebook GPU with only 4 GB of video RAM[8].

The loss function is calculated from adding the losses from the predicting the centre of the bounding box (x, y) with the losses from predicting width and height of that bounding box w and h, the losses from predicting the confidence and the losses from predicting the right class. For calculating the losses binary cross-entropy with logits, the logarithm of the probabilities [63] is used. The losses are masked, meaning that they are only calculated for the corresponding output layer and are therefore only used to minimise the error for this particular output layer. To update the weights of the convolutional neural networks in the model, Adam [42] is used as optimiser with a learning rate of 10^{-3}.

Predicting bounding boxes for a given input image is again fairly standard: first the size of

[6] The whole process is further elaborated in section 3.4.1.

[7] Which is explained in section 3.3.

[8] In this case it was an NVIDIA GeForce GTX 1050 Ti with 4 GB of VRAM.

the image is adjusted, so that it fits to the input layer of the convolutional neural networks, where the aspect ratio of the image is left unchanged using padding. Then an additional, but empty dimension is added to the input image data, to ensure that the input data has the same dimensionality as the inputs for the training data, which had one extra dimension to enable batch learning of the training examples. Afterwards the system predicts the location of the bounding boxes, the confidence scores for these predictions and the class to which the object belongs to. Since in this system there is only one class, whether a detected object is a solar panel, this value gets discarded. The confidentiality threshold is set to 0.55, to minimise the amount of false positives and make sure that the system only returns objects, where it is rather sure that they are solar panels.

The code of my implementation will be provided on GitHub [65].

3.4.1 Obtaining and preprocessing the Data

One of the biggest challenges in creating this system was to obtain and manage the large amounts of data for training and using the system. The full training data set [11] consists of 601 images, which make a total of 44.8 GB. For training this system the data from the city of Oxnard was excluded, due to the different size of those images, making the preprocessing more difficult. In total 526 images (or 39.4 GB) were used for training. The original size of the training data was 5000×5000 pixels the training images were split into images of 1000×1000 pixels. Training the YOLO algorithm on the original sizes works, but the detection performance is much better for the smaller images, since they are much closer to the resolution used by the input layer of the convolutional neural networks (416×416) and therefore fewer details get lost in the process of downscaling the input images. The algorithm was trained using only images in which solar panels are present, which were 3145 images from the cities of Modesto, Stockton and Fresno or 10.3 GB of training data. While splitting the input images, also the coordinates of the solar panels in the images are adjusted accordingly. Another step during this preprocessing of the input images is to convert the locations of the solar panels in the images from polygons to rectangular bounding boxes. The results of this process are stored to a text file containing the full path of an input image and the coordinates for all bounding boxes in this image together with the class identification[9], which is then loaded by the system during the training phase.

After the training is done, the system was used on high resolution aerial image data provided from the city of Vienna [93]. The city of Vienna provides this data via the Web Map Tile Service (WMTS) format [54], an open standard for providing tiles of geoinformation data over the web.

[9]Which is always the same, since there is only one class to identify.

To access and download this data, I used the Python bindings for GDAL [79], an open source library for accessing various raster and vector geospatial data formats. GDAL allows not only to access local resources like files, but also to access online data provided by services like WMTS. To download files from the WMTS server I used the method `gdal.Translate()`, a general purpose function for translating from one geoinformation format to another. When downloading, the system sends queries for the given layer with a window size of 1000×1000 to the WMTS server until it reaches the maximum values for the x- and y-coordinates. To speed up downloading, 16 downloads are performed in parallel, this is a compromise of speeding up the download and not flooding the server with requests until it refuses to connect with the downloading client. Since the download process can still be unsuccessful from time to time, the system tries to download the whole data five times, but already downloaded files are not overwritten. For using the system, the high resolution aerial image data from 2017 is downloaded. Downloading the files tile-wise comes with the advantage, that no further preprocessing is necessary when using the data, if the whole image for one layer would be downloaded it would be 152860×128690 pixel tall and it would be 78.7 GB of size, downloading the data as tiles results in 19542 files with a total size of 77.9 GB, the difference in size comes from not downloading tiles of the edges. Even though it is not needed, GDAL would be able to handle an image files with several gigabytes in filesize.

3.5 Evaluation of the Results

Figure 3.5 shows the loss of the model during training. During the first epochs the loss is rather high, but it quickly drops, suggesting that the model learns to better identify solar panels in the training data. After 25 epochs, the model reaches a plateau where longer training doesn't lead to much improvement of the model's performance. The model uses an adaptive learning rate, that automatically adjusts itself when reaching a plateau to not miss a minimum that otherwise might be overlooked, if a higher learning rate is used. Figure 3.6 shows how the learning rate decreases over time, this decrease is analogous to the model reaching the plateau from epoch 25 on. From that epoch on the learning rate gets decreased in every third subsequent step, this was set when creating the model, it shows that the model reached a local minimum (and most likely a global minimum as well) of its loss function.

The model is able to correctly identify two solar panels in these images. The large solar panel in the upper right part of image 3.7b is identified with a rather low confidence of 0.57, even though for a human it is quite obvious that it is a solar panel. In the same picture there is an array of photovoltaic systems, the model is able to identify one solar panel from this array with a high confidence of 0.7, but at the same time it is not able to identify the whole array. In image 3.7a, there is a similar situation in the upper right part: the model is able to identify the two solar panels that belong to the same photovoltaic array, but it can not identify them as belonging together. The yellow boxes in the image show objects which look very similar to solar panels, but were not identified as such. Also, for a human it is very difficult to identify those objects as solar panels or not. I would say that the region encapsulated by a yellow box in image 3.7a is a solar panel, while the object encapsulated by a yellow box in image 3.7b is just a shadow. The model is able to detect all solar panels in picture 3.8a, but two of them are actually shadows, only the one in the lower left corner with a confidence of 0.78 is an actual solar panel. Similarly, in picture 3.8b, there is actually no solar panel, but the model identifies a shadow as solar panel. It has only 0.59 confidence in this prediction, but this value is above the threshold of 0.55, set by me and therefore the model reports it. I chose a threshold value of 0.55 to have an exclusion criteria for false positives, the value was chosen to strike a balance between excluding false positives, but not excluding true positives. Achieving such a balance is a rather difficult task, for example the solar panel in the upper part of image 3.7b only has a confidence value of 0.57, while the shadow in the lower part of image 3.8b has a confidence score of 0.59.

Figure 3.9 shows a histogram of the confidence scores when applying the model to high resolution aerial data from Vienna. The histogram consists of nine bins, where each bin has a size of 0.05. In total the model finds 7896 solar panels, table 3.1 shows the exact number of objects falling into one bin of that histogram. The total number of objects detected by the algorithm is heavily dependent on the confidence threshold value for objects being detected by the system. As mentioned above, so far this value is set to 0.55, but this value might changes in future iterations of this work.

3.6 Future work

The system in its current state is still work-in-progress and so far a good basis for further development. There is a lot of room for improvement, especially when it comes to tuning hyperparameters or to finding a threshold value that represents a good middle-ground between ignoring false positives and not including, what until now, is classified as solar panel, but is actually a false negatives. Due to time constraints I was compelled to stop adding features to the system. One such feature was adding different machine learning algorithms and compare their performance with each other. Since YOLO is a deep learning algorithm, it would be interesting to compare its performance on this data set to traditional machine learning algorithms like

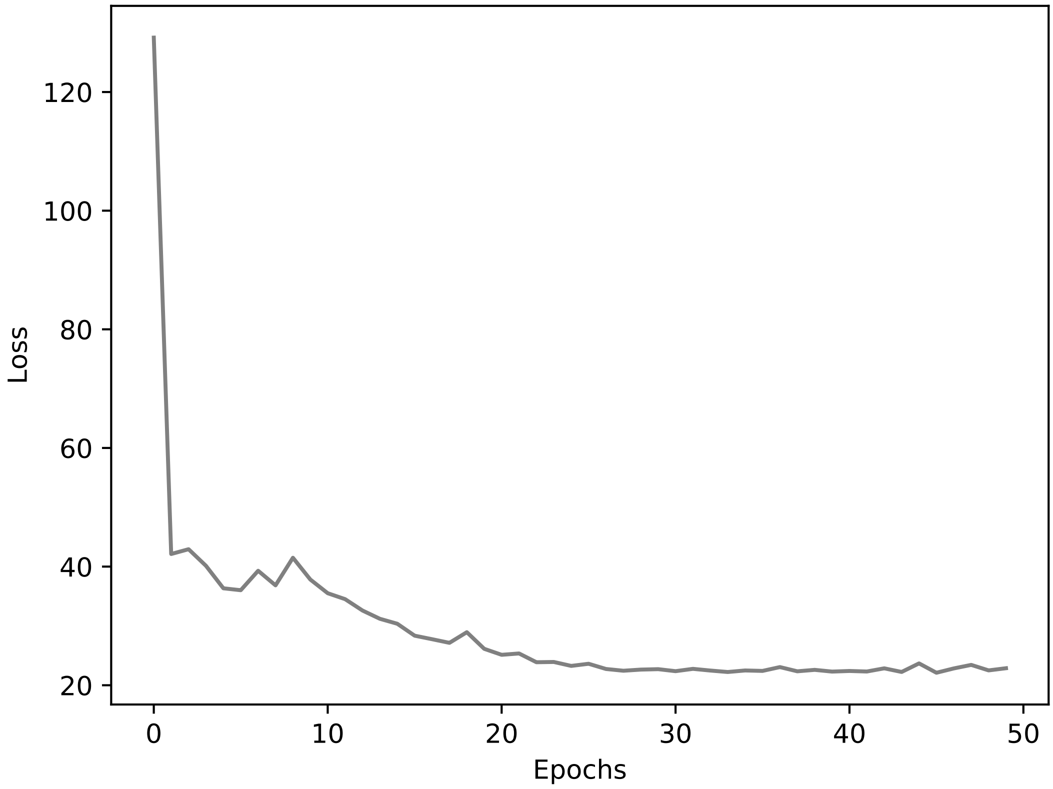


Figure 3.5: Loss function during training period. The loss value rapidly drops after the first epochs, then it further decreases until around 30 epochs, from then on the loss keeps plateauing with some minor spikes, but ultimately the loss stays at values between 22 and 23.

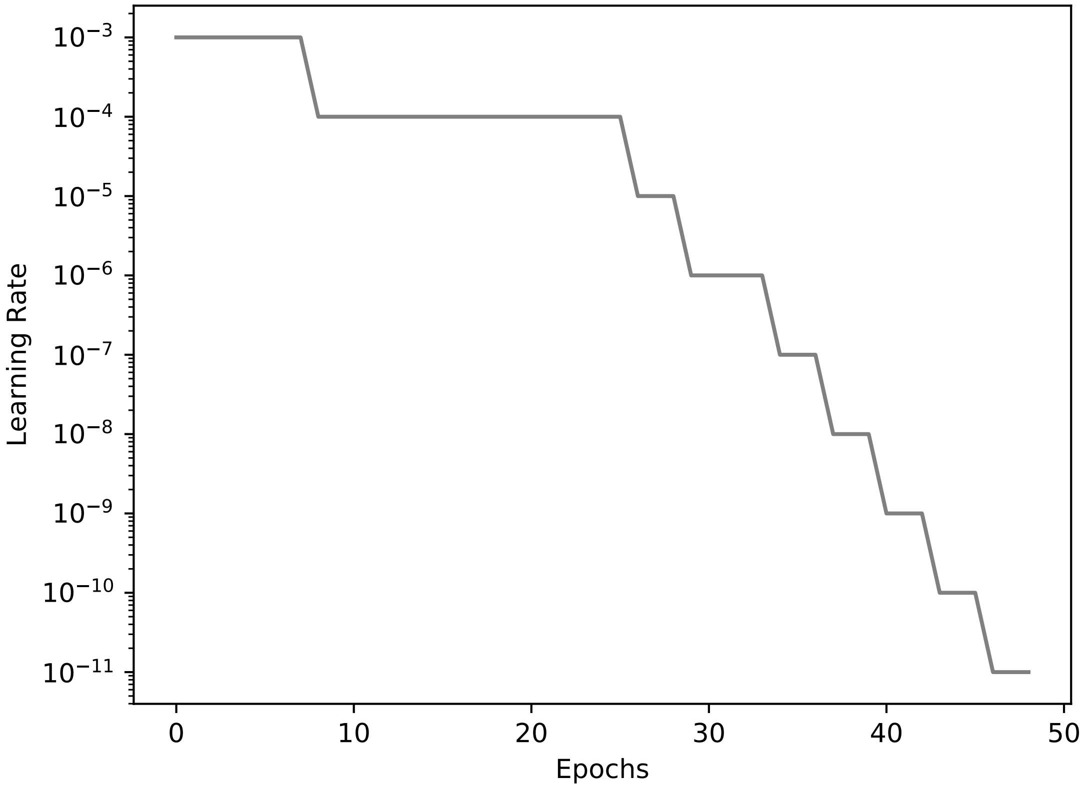


Figure 3.6: Change of learning rate during training. The model uses an adaptive learning rate. When the loss reaches a plateau, the learning rate gets decreased to find a possible minimum on the plateau. After 8 epochs, the learning gets adjusted for the first time, by multiplying it by a factor of 0.1. This happens again after epoch 26 and from that on every three epochs. The y-axis for this plot is using a logarithmic scale, to better see the reductions of the learning rate.

Bin start	Number of objects
0.55	1363
0.60	1252
0.65	1069
0.70	948
0.75	867
0.80	812
0.85	745
0.90	572
0.96	268

Table 3.1: Values for the histogram shown in figure 3.9.

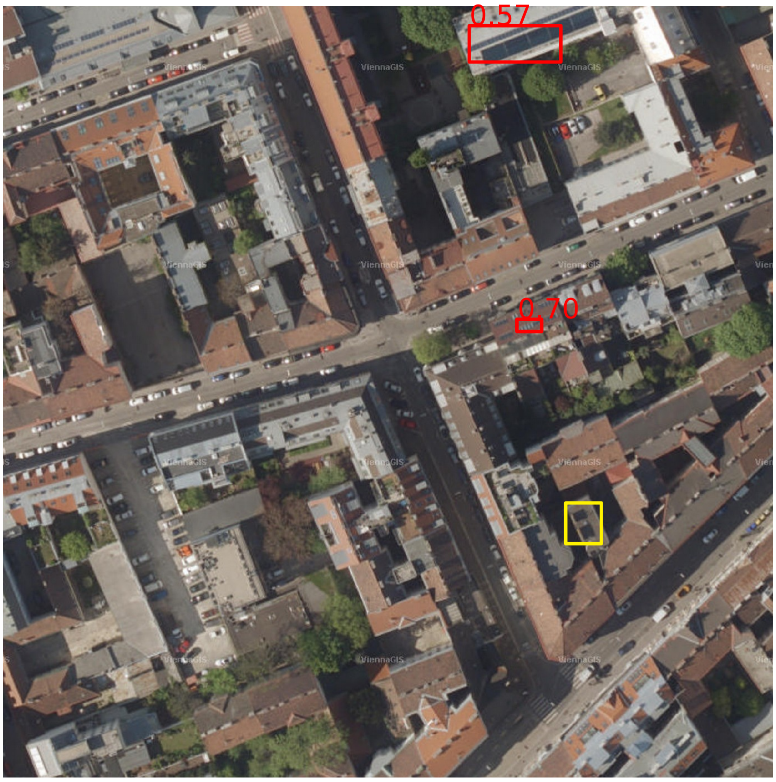
support vector machines or random forest classifiers. Random forest classifiers were used in [52], it would be interesting to see how their performance is on high resolution aerial image data from Vienna. Similarly, it would be interesting to compare the performance of different deep learning algorithms for object detection like different versions of Regions with Convolutional Neural Networks (R-CNN) or Spatial Pyramid Pooling in Deep Convolutional Networks (SPP-net) [100] on this data set.

Decreasing the size of the used images from currently 1000×1000 to 500×500 could also improve the performance of the system, since the images for training and for detecting solar panels in Vienna are closer to the native resolution used for the convolutional neural networks of YOLO (416×416) and therefore have to be scaled less. The downside of this is the increased time it takes to learn from the increased amount of training examples. As written above, training data is only used from three of the four available cities. When reducing the image size, it would also be easier to integrate the remaining training data for Oxnard, which would result in more training data and therefore might increase the object detection performance of the system, but at the same time increases the time needed for training.

Verifying the results from Vienna is rather difficult, since it has to be done manually. One way to speed this process up is to use the available data for subsidised photovoltaic systems in Vienna [92] and compare the results from the system with this data. It is not a perfect metric, since this data for this data set is itself not exhaustive, but it is a good starting point and reduces the effort of manually validating the results. Due to the time-consuming nature of evaluating the results, further evaluation and analysis of the results are points were future work in this direction could continue. Especially evaluating how the number of solar panels has changed in Vienna from 2014 to 2017. So far this system was only tested on data from Vienna, another point that could be tackled in future work is using the system on high resolution aerial images from different cities, towns, villages or for any kind of high resolution aerial image data.

Finally, the visualisation of the results could be greatly improved. So far the results are only shown locally, on the image used for detection. Those images have geospatial information and therefore the bounding boxes drawn onto these images can quite easily be transferred to and visualised using other geographic information systems like QGIS [80], OpenStreetMap [19] or Google Maps [47].

(a) Example where the model detects two solar panels, which turn out to be true positives



(b) Example where the model detects two solar panels, which turn out to be true positives

Figure 3.7: Examples of true positives from high resolution aerial images of Vienna. Images taken from [93].

(a) Example were the model detects three solar panels, but only one of them turns out to be a true positive.

(b) Example were the model detects on solar panel, which turns out to be a false positive.

Figure 3.8: Examples of false positives from high resolution aerial images of Vienna. Images taken from [93].

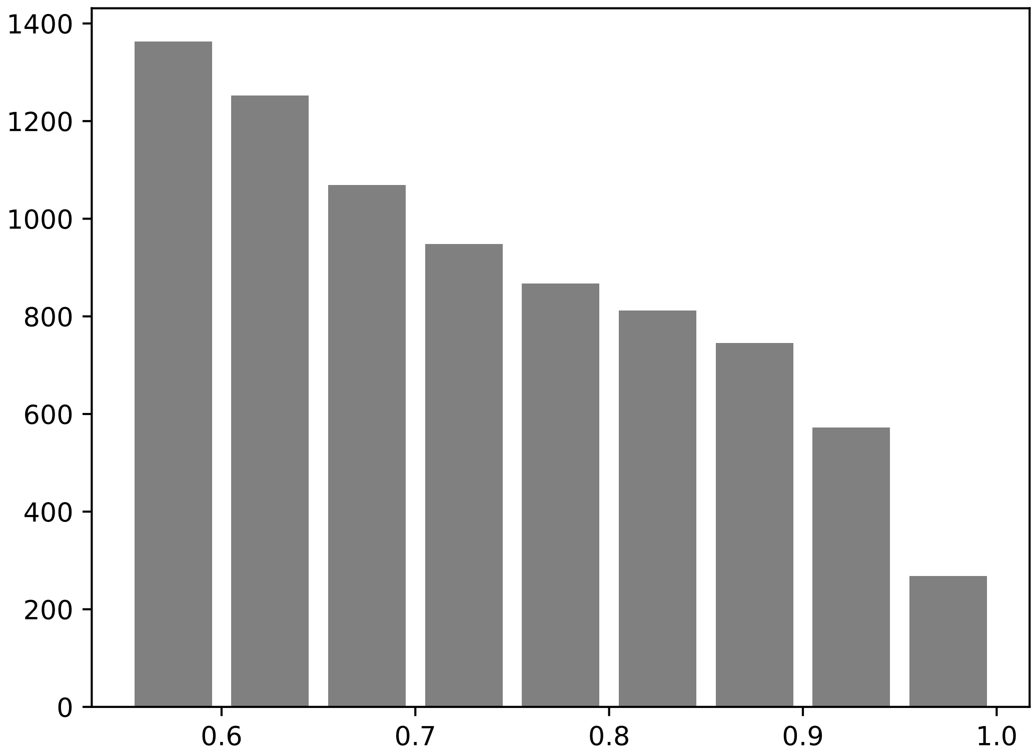


Figure 3.9: Histogram of the confidence scores for running the model on high resolution aerial image data of Vienna from 2017. The size of one bin is 0.5 and there are 9 bins in total. In total the model detected 7896 objects, the confidence scores for these detections are quite evenly distributed, only scores where the model is very sure (≥ 0.95) are rather rare.

Chapter 4

Conclusion

This thesis was split into two parts, which approached social energy systems from different angles. The first part studies microgrids and introduces the concept of synchronisation games, where selfish prosumers cooperate to artificially drive up the load of a microgrid and profit of the higher prices that come with the higher load, since it allows them to sell the energy they generate for a higher price. The second part proposes a system to automatically detect photovoltaic arrays from high resolution aerial images. This system is already able to detect solar panels, but so far it is still more a proof-of-concept than a system which can be used in practice. The work done so far will be used as basis for further work in this field. Besides overcoming current technical challenges, using the system for more data than just from the city of Vienna are the top priorities on the roadmap for the further development of this system. The task of detecting solar panels from images is a rather challenging one, the top contestants in a Kaggle challenge for detecting solar panels from orthoimagery reached a score of less than 80% [40].

This work shows that energy systems are becoming more and more "social", due to new developments in energy generation and distribution. The shift from big, centralised entities that generate energy towards smaller, decentralised facilities for generating energy makes the whole power grid more social, since the whole process and everything that comes with it are not just a one-to-many relationship any more, but a many-to-many relationship. It is in some regards similar to the rise of the internet, which allowed anyone to not only be a passive media consumer, but to also take an active role and to create content. Just like the democratisation of media with the rise of the internet brought opportunities, benefits and challenges, the democratisation of the energy sector will bring the same.

Bibliography

[1] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/` (visited on 25th Nov. 2018).

[2] Vikram Aggarwal. *How much energy does a solar panel actually produce? Electricity output explained.* 2018. URL: `https://news.energysage.com/what-is-the-power-output-of-a-solar-panel/` (visited on 14th Nov. 2018).

[3] Jose F. Alfaro et al. 'Improving rural electricity system planning: An agent-based model for stakeholder engagement and decision making'. In: *Energy Policy* 101 (2017), pp. 317–331. ISSN: 0301-4215. DOI: `10.1016/j.enpol.2016.10.020`. URL: `http://www.sciencedirect.com/science/article/pii/S030142151630564X`.

[4] Moudhi M. Aljamea et al. 'Smart Meter Data Analysis'. In: *Proceedings of the International Conference on Internet of Things and Cloud Computing.* ICC '16. Cambridge, United Kingdom: ACM, 2016, 22:1–22:6. ISBN: 978-1-4503-4063-2. DOI: `10.1145/2896387.2896407`.

[5] Anonymous. *keras-yolo3. A Keras implementation of YOLOv3 (Tensorflow backend).* GitHub Repository. 2018. URL: `https://github.com/qqwweee/keras-yolo3` (visited on 25th Nov. 2018).

[6] Bundesverband Photovoltaic Austria. *Photovoltaik-Förderung in Österreich.* 2018. URL: `https://www.pvaustria.at/forderungen/` (visited on 19th Nov. 2018).

[7] Emmanuel P. Baltsavias. 'Object extraction and revision by image analysis using existing geodata and knowledge: current status and steps towards operational systems'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 58.3 (2004). Integration of Geodata and Imagery for Automated Refinement and Update of Spatial Databases, pp. 129–151. ISSN: 0924-2716. DOI: `10.1016/j.isprsjprs.2003.09.002`. URL: `http://www.sciencedirect.com/science/article/pii/S0924271603000546`.

[8] I. Safak Bayram et al. 'A survey on energy trading in smart grid'. In: *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Dec. 2014, pp. 258–262. DOI: 10.1109/globalsip.2014.7032118.

[9] Peter Biermayr et al. *Innovative Energietechnologien in Österreich Marktentwicklung 2014*. 2015. URL: https://www.pvaustria.at/wp-content/uploads/Marktstatistik_2014-Final.pdf (visited on 19th Nov. 2018).

[10] Thomas Blaschke. 'Object based image analysis for remote sensing'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 65.1 (2010), pp. 2–16. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2009.06.004. URL: http://www.sciencedirect.com/science/article/pii/S0924271609000884.

[11] Kyle Bradbury et al. 'Distributed solar photovoltaic array location and extent dataset for remote sensing object identification'. In: *Scientific Data* 3 (Dec. 2016). Data Descriptor. DOI: 10.1038/sdata.2016.106.

[12] Steven Brown. *Building a Smart PySC2 Agent*. 2017. URL: https://chatbotslife.com/building-a-smart-pysc2-agent-cdc269cb095d (visited on 18th Nov. 2018).

[13] Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen. *Registrierung von Photovoltaikanlagen im Marktstammdatenregister*. 2018. URL: https://www.bundesnetzagentur.de/DE/Sachgebiete/ElektrizitaetundGas/Unternehmen_Institutionen/DatenaustauschundMonitoring/MaStR/RegistrPVAnlagen/RegistrPVAnlagen_node.html (visited on 3rd Dec. 2018).

[14] Joseph Camilo et al. 'Application of a semantic segmentation convolutional neural network for accurate automatic detection and mapping of solar photovoltaic arrays in aerial imagery'. In: *arXiv preprint arXiv:1801.04018* (2018).

[15] National Energy Board Canada. *Provincial and Territorial Energy Profiles – New Brunswick*. 2018. URL: https://www.neb-one.gc.ca/nrg/ntgrtd/mrkt/nrgsstmprfls/nb-eng.html?=undefined&wbdisable=true (visited on 14th Nov. 2018).

[16] Debasis Chaudhuri et al. 'Automatic Building Detection From High-Resolution Satellite Images Based on Morphology and Internal Gray Variance'. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 9.5 (May 2016), pp. 1767–1779. ISSN: 1939-1404. DOI: 10.1109/JSTARS.2015.2425655.

[17] Xueyun Chen et al. 'Vehicle Detection in Satellite Images by Parallel Deep Convolutional Neural Networks'. In: *2013 2nd IAPR Asian Conference on Pattern Recognition*. Nov. 2013, pp. 181–185. DOI: 10.1109/ACPR.2013.33.

[18] François Chollet et al. *Keras*. 2015. URL: https://keras.io (visited on 25th Nov. 2018).

[19] OpenStreetMap Community. *OpenStreetMap*. 2004. URL: https://www.openstreetmap.org (visited on 28th Nov. 2018).

[20] Leeds City Council. *Solar PV (photovolataics) data*. 2016. URL: https://datamillnorth.org/dataset/solar-pv--photovolataics--data (visited on 3rd Dec. 2018).

[21] Luke Daniel. *All solar panels in Cape Town to be registered by 2019*. 2018. URL: https://www.thesouthafrican.com/all-solar-panels-in-cape-town-to-be-registered-by-2019/ (visited on 3rd Dec. 2018).

[22] Luke Daniel. *New Cape Town by-law to fine unregistered solar panel owners*. 2018. URL: https://www.thesouthafrican.com/cape-town-solar-panel-how-to-register/ (visited on 3rd Dec. 2018).

[23] Bundesministerium für Digitalisierung und Wirtschaftsstandort. *Gesamte Rechtsvorschrift für NÖ Bauordnung 2014, Fassung vom 03.12.2018*. 2018. URL: https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=LrNO&Gesetzesnummer=20001079 (visited on 3rd Dec. 2018).

[24] Bundesministerium für Digitalisierung und Wirtschaftsstandort. *Gesamte Rechtsvorschrift für Steiermärkisches Baugesetz, Fassung vom 03.12.2018*. 2018. URL: https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=LrStmk&Gesetzesnummer=20000070 (visited on 3rd Dec. 2018).

[25] Cherrelle Eid et al. 'Time-based pricing and electricity demand response: Existing barriers and next steps'. In: *Utilities Policy* 40 (2016), pp. 15–25. ISSN: 0957-1787. DOI: 10.1016/j.jup.2016.04.001. URL: http://www.sciencedirect.com/science/article/pii/S0957178716300947.

[26] Department of Energy. *The Open PV Project*. 2016. URL: https://catalog.data.gov/dataset/the-open-pv-project (visited on 3rd Dec. 2018).

[27] Mark Everingham et al. 'The Pascal Visual Object Classes (VOC) Challenge'. In: *International Journal of Computer Vision* 88.2 (June 2010), pp. 303–338.

[28] M. Nazif Faqiry et al. 'Game theoretic model of energy trading strategies at equilibrium in microgrids'. In: *2014 North American Power Symposium (NAPS)*. Sept. 2014, pp. 1–4. DOI: 10.1109/NAPS.2014.6965413.

[29] Michael J. Fell and David Shipworth. ‘Behaviour, practice–whatever? A theory-agnostic framework for describing and informing demand-side response’. In: *ECEEE Summer Study 2017 Proceedings*. European Council for an Energy Efficient Economy (ECEEE). 2017, pp. 1603–1612.

[30] Energy Web Foundation. *The Decentralized Autonomous Area Agent (D3A) Market Model*. 2018. URL: `https://energyweb.org/wp-content/uploads/2018/04/EWF-D3A-ConceptBrief-FINAL201804.pdf` (visited on 14th Nov. 2018).

[31] King-Sun Fu and Azriel Rosenfeld. ‘Pattern Recognition and Image Processing’. In: *IEEE Transactions on Computers* C-25.12 (Dec. 1976), pp. 1336–1346. ISSN: 0018-9340. DOI: `10.1109/TC.1976.1674602`.

[32] Simon Gächter. ‘Behavioral game theory’. In: *Blackwell handbook of judgment and decision making*. Ed. by Derek J. Koehler and Nigel Harvey. Blackwell: Oxford, UK, 2004. Chap. 24, pp. 485–503.

[33] Saman Ghaffarian and Salar Ghaffarian. ‘Automatic building detection based on Purposive FastICA (PFICA) algorithm using monocular high resolution Google Earth images’. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 97 (2014), pp. 152–159. ISSN: 0924-2716. DOI: `10.1016/j.isprsjprs.2014.08.017`. URL: `http://www.sciencedirect.com/science/article/pii/S0924271614002238`.

[34] Grid Singularity - GSy GmbH. *Grid Singularity*. 2018. URL: `https://gridsingularity.com/` (visited on 14th Nov. 2018).

[35] Nikos Hatziargyriou. *Microgrids: architectures and control*. John Wiley & Sons, 2014.

[36] Clara F. Heuberger et al. ‘A systems approach to quantifying the value of power generation and energy storage technologies in future electricity networks’. In: *Computers & Chemical Engineering* 107 (2017), pp. 247–256. ISSN: 0098-1354. DOI: `10.1016/j.compchemeng.2017.05.012`. URL: `http://www.sciencedirect.com/science/article/pii/S0098135417302119`.

[37] Tyler Hodge and Carol A. Dahl. ‘Power marketer pricing behavior in the California Power Exchange’. In: *Energy Economics* 34.2 (2012), pp. 568–575. ISSN: 0140-9883. DOI: `10.1016/j.eneco.2011.05.003`. URL: `http://www.sciencedirect.com/science/article/pii/S0140988311001083`.

[38] Richard Hodson. ‘Climate change’. In: *Nature* 550 (Oct. 2017), S53 EP -. DOI: `10.1038/550S53a`.

[39] Bohao Huang et al. 'Dense labeling of large remote sensing imagery with convolutional neural networks: a simple and faster alternative to stitching output label maps'. In: *arXiv preprint arXiv:1805.12219* (2018).

[40] Kaggle Inc. *Solar Array Object Detection in Orthoimagery*. 2016. URL: https://www.kaggle.com/c/solar-array-object-detection-in-orthoimagery/leaderboard (visited on 6th Dec. 2018).

[41] Aniruddha Kembhavi, David Harwood and Larry S. Davis. 'Vehicle Detection Using Partial Least Squares'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.6 (June 2011), pp. 1250–1265. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2010.182.

[42] Diederik P. Kingma and Jimmy L. Ba. 'Adam: A method for stochastic optimization'. In: *arXiv preprint arXiv:1412.6980* (2014).

[43] Elizaveta Kuznetsova et al. 'An integrated framework of agent-based modelling and robust optimization for microgrid energy management'. In: *Applied Energy* 129 (2014), pp. 70–88. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2014.04.024. URL: http://www.sciencedirect.com/science/article/pii/S0306261914003766.

[44] Joohyung Lee, Jun Guo and Jun Kyun Choi Moshe Zukerman. 'Distributed Energy Trading in Microgrids: A Game-Theoretic Model and Its Equilibrium Analysis'. In: *IEEE Transactions on Industrial Electronics* 62.6 (June 2015), pp. 3524–3533. ISSN: 0278-0046. DOI: 10.1109/TIE.2014.2387340.

[45] Chaddenwych Services Limited. *Electron*. 2016. URL: http://www.electron.org.uk/ (visited on 14th Nov. 2018).

[46] Tsung-Yi Lin et al. 'Feature Pyramid Networks for Object Detection'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017, pp. 936–944. DOI: 10.1109/CVPR.2017.106.

[47] Google LLC. *Google Maps*. 2005. URL: https://www.google.com/maps (visited on 28th Nov. 2018).

[48] Power Ledger Pty Ltd. *Power Ledger white paper*. 2017. URL: https://cdn2.hubspot.net/hubfs/4519667/Documents%20/Power%20Ledger%20Whitepaper.pdf (visited on 14th Nov. 2018).

[49] Satish Madhogaria et al. 'Car detection by fusion of HOG and causal MRF'. In: *IEEE Transactions on Aerospace and Electronic Systems* 51.1 (Jan. 2015), pp. 575–590. ISSN: 0018-9251. DOI: 10.1109/TAES.2014.120141.

[50] Jordan M. Malof, Leslie M. Collins and Kyle Bradbury. 'A deep convolutional neural network, with pre-training, for solar photovoltaic array detection in aerial imagery'. In: *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. July 2017, pp. 874–877. DOI: 10.1109/IGARSS.2017.8127092.

[51] Jordan M. Malof et al. 'A deep convolutional neural network and a random forest classifier for solar photovoltaic array detection in aerial imagery'. In: *2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA)*. Nov. 2016, pp. 650–654. DOI: 10.1109/ICRERA.2016.7884415.

[52] Jordan M. Malof et al. 'Automatic detection of solar photovoltaic arrays in high resolution aerial imagery'. In: *Applied energy* 183 (2016), pp. 229–240.

[53] Jordan M. Malof et al. 'Image features for pixel-wise detection of solar photovoltaic arrays in aerial imagery using a random forest classifier'. In: *2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA)*. Nov. 2016, pp. 799–803. DOI: 10.1109/ICRERA.2016.7884446.

[54] Joan Maso, Keith Pomakis and Nuria Julia. 'OpenGIS web map tile service implementation standard'. In: *Open Geospatial Consortium Inc* (2010), pp. 4–6.

[55] Francisco P. Maturana et al. 'Agent-based testbed simulator for power grid modeling and control'. In: *2012 IEEE Energytech*. May 2012, pp. 1–6. DOI: 10.1109/EnergyTech.2012.6304661.

[56] Brooklyn Microgrid. *Brooklyn Microgrid*. 2016. URL: https://www.brooklyn.energy/ (visited on 6th Dec. 2018).

[57] David L. Mills. *Computer network time synchronization: the network time protocol on earth and in space*. CRC Press, 2006.

[58] Hiroki Miyamoto et al. 'Object Detection in Satellite Imagery Using 2-Step Convolutional Neural Networks'. In: *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*. July 2018, pp. 1268–1271. DOI: 10.1109/IGARSS.2018.8518587.

[59] Volodymyr Mnih et al. 'Playing atari with deep reinforcement learning'. In: *arXiv preprint arXiv:1312.5602* (2013).

[60] John Nash. 'Non-cooperative games'. In: *Annals of mathematics* (1951), pp. 286–295.

[61] David Newbery. 'Electricity liberalisation in Britain: The quest for a satisfactory wholesale market design'. In: *The Energy Journal* 26 (2005), pp. 43–70. ISSN: 01956574, 19449089. URL: http://www.jstor.org/stable/23297006.

[62] Land Niederösterreich. *PV-Anlagen in NÖ*. 2018. URL: https://www.data.gv.at/katalog/dataset/pv-anlagen-in-no (visited on 19th Nov. 2018).

[63] Sharyn O'Halloran. *Lecture 9: Logit/Probit*. 2010. URL: http://www.columbia.edu/~so33/SusDev/Lecture_9.pdf (visited on 27th Nov. 2018).

[64] Lawrence Orsini. 'Use of blockchain based distributed consensus control'. 16/015,845. 22nd June 2018.

[65] Franz Papst. *Solar Panel Detection*. GitHub Repository. 2018. URL: https://github.com/CPUFronz/solar_panel_detection (visited on 4th Dec. 2018).

[66] Larry Partain et al. ''Swanson's Law' plan to mitigate global climate change'. In: *2016 IEEE 43rd Photovoltaic Specialists Conference (PVSC)*. June 2016, pp. 3335–3340. DOI: 10.1109/PVSC.2016.7750284.

[67] Matthias Pilz and Luluwah Al-Fagih. 'Game-Theoretic Approaches to Energy Trading: A Survey'. In: *CoRR* (2017).

[68] Elizabeth L. Ratnam et al. 'Residential load and rooftop PV generation: an Australian distribution network dataset'. In: *International Journal of Sustainable Energy* 36.8 (2017), pp. 787–806. DOI: 10.1080/14786451.2015.1100196.

[69] Joseph Redmon. *Darknet: Open Source Neural Networks in C*. 2013–2016. URL: http://pjreddie.com/darknet/ (visited on 25th Nov. 2018).

[70] Joseph Redmon and Ali Farhadi. 'YOLO9000: better, faster, stronger'. In: *arXiv preprint arXiv:1612.08242* (2016).

[71] Joseph Redmon and Ali Farhadi. 'Yolov3: An incremental improvement'. In: *arXiv preprint arXiv:1804.02767* (2018).

[72] Joseph Redmon et al. 'You Only Look Once: Unified, Real-Time Object Detection'. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[73] Philipp Ringler, Dogan Keles and Wolf Fichtner. 'Agent-based modelling and simulation of smart electricity grids and markets – A literature review'. In: *Renewable and Sustainable Energy Reviews* 57 (2016), pp. 205–215. ISSN: 1364-0321. DOI: 10.1016/j.rser.2015.12.169. URL: http://www.sciencedirect.com/science/article/pii/S136403211501552X.

[74] Olga Russakovsky et al. 'ImageNet Large Scale Visual Recognition Challenge'. In: *International Journal of Computer Vision* 115.3 (Dec. 2015), pp. 211–252. ISSN: 1573-1405. DOI: 10.1007/s11263-015-0816-y.

[75] Olga Saukh, Franz Papst and Sergii Saukh. 'Synchronization Games in P2P Energy Trading'. English. In: *Proceedings of the IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*. Oct. 2018.

[76] Steven Stoft. 'Power system economics'. In: *Journal of Energy Literature* 8 (2002), pp. 94–99.

[77] SunContract. *SunContract White Paper*. 2017. URL: `https://suncontract.org/res/whitepaper.pdf` (visited on 14th Nov. 2018).

[78] Richard M. Swanson. 'A vision for crystalline silicon photovoltaics'. In: *Progress in Photovoltaics: Research and Applications* 14.5 (2006), pp. 443–453. DOI: `10.1002/pip.709`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/pip.709`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/pip.709`.

[79] GDAL Development Team. *GDAL - Geospatial Data Abstraction Library, Version 2.2.3.3*. Open Source Geospatial Foundation, 2018. URL: `http://www.gdal.org` (visited on 27th Nov. 2018).

[80] QGIS Development Team. *QGIS Geographic Information System. Open Source Geospatial Foundation*. 2009. URL: `https://www.qgis.org/` (visited on 28th Nov. 2018).

[81] Alvin Toffler. *The third wave*. Vol. 484. Bantam books New York, 1980.

[82] Jasna Tomić and Willett Kempton. 'Using fleets of electric-drive vehicles for grid support'. In: *Journal of Power Sources* 168.2 (2007), pp. 459–468. ISSN: 0378-7753. DOI: `10.1016/j.jpowsour.2007.03.010`. URL: `http://www.sciencedirect.com/science/article/pii/S0378775307005575`.

[83] Alexander Toshev, Ben Taskar and Kostas Daniilidis. 'Shape-Based Object Detection via Boundary Structure Segmentation'. In: *International Journal of Computer Vision* 99.2 (Sept. 2012), pp. 123–146. ISSN: 1573-1405. DOI: `10.1007/s11263-012-0521-z`.

[84] Walter R. Tribett et al. 'Paris INDCs'. In: *Paris Climate Agreement: Beacon of Hope*. Cham: Springer International Publishing, 2017, pp. 115–146. ISBN: 978-3-319-46939-3. DOI: `10.1007/978-3-319-46939-3_3`.

[85] Matthew A. Turk and Alex P. Pentland. 'Face recognition using eigenfaces'. In: *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. June 1991, pp. 586–591. DOI: `10.1109/CVPR.1991.139758`.

[86] Klaus Wallnöfer. 'Die Energiewirtschaft vor dem zweiten Umbruch?' In: *Wirtschaftsrechtliche Blätter* 22.1 (Jan. 2008), pp. 13–20. ISSN: 0930-3855. DOI: `10.1007/s00718-007-1042-3`.

[87] Huiwei Wang et al. 'Reinforcement Learning in Energy Trading Game Among Smart Microgrids'. In: *IEEE Transactions on Industrial Electronics* 63.8 (Aug. 2016), pp. 5109–5119. ISSN: 0278-0046. DOI: 10.1109/TIE.2016.2554079.

[88] Min Wang et al. 'Photovoltaic panel extraction from very high-resolution aerial imagery using region–line primitive association analysis and template matching'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 141 (2018), pp. 100–111. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2018.04.010. URL: http://www.sciencedirect.com/science/article/pii/S0924271618301205.

[89] Rui Wang et al. 'The poor generalization of deep convolutional networks to aerial imagery from new geographic locations: an empirical study with solar array detection'. In: *2017 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*. Vol. 00. Oct. 2017, pp. 1–8. DOI: 10.1109/AIPR.2017.8457965.

[90] Christopher J. C. H. Watkins. 'Learning from delayed rewards'. PhD thesis. King's College, Cambridge, 1989.

[91] Christopher J. C. H. Watkins and Peter Dayan. 'Q-learning'. In: *Machine Learning* 8.3 (May 1992), pp. 279–292. ISSN: 1573-0565. DOI: 10.1007/BF00992698.

[92] Stadt Wien. *Geförderte Photovoltaikanlagen Standorte Wien*. 2018. URL: https://www.data.gv.at/katalog/dataset/7fb52cf2-2222-4370-b9df-43464b26a043 (visited on 19th Nov. 2018).

[93] Stadt Wien. *Web Map Tile Service (WMTS) Wien*. 2018. URL: https://www.data.gv.at/katalog/dataset/stadt-wien_webmaptileservicewmtswien (visited on 27th Nov. 2018).

[94] Uri Wilensky and William Rand. *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo*. MIT Press, 2015.

[95] Elizabeth Woyke. *Brooklyn Microgrid*. 2017. URL: https://www.technologyreview.com/s/604227/blockchain-is-helping-to-build-a-new-kind-of-energy-grid/ (visited on 6th Dec. 2018).

[96] Liang Xiao et al. 'Reinforcement Learning-based Energy Trading for Microgrids'. In: *arXiv preprint arXiv:1801.06285* (2018).

[97] Mahmoud Mohammed Sidi Youssef et al. 'Combining top-down and bottom-up approaches for building detection in a single very high resolution satellite image'. In: *2014 IEEE Geoscience and Remote Sensing Symposium*. July 2014, pp. 4820–4823. DOI: 10.1109/IGARSS.2014.6947573.

[98] Tarek A. Youssef et al. 'GPS Synchronization of Smart Distributed Converters for Microgrid Applications'. In: *Energies* 11.4 (2018). ISSN: 1996-1073. DOI: 10.3390/en11040695. URL: http://www.mdpi.com/1996-1073/11/4/695.

[99] Damin Zhang et al. 'Aerial image analysis based on improved adaptive clustering for photovoltaic module inspection'. In: *2017 International Smart Cities Conference (ISC2)*. Sept. 2017, pp. 1–6. DOI: 10.1109/ISC2.2017.8090798.

[100] Zhong-Qiu Zhao et al. 'Object detection with deep learning: A review'. In: *arXiv preprint arXiv:1807.05511* (2018).

[101] Mofan Zhou. *Reinforcement Learning Methods and Tutorials*. GitHub Repository. 2017. URL: https://github.com/MorvanZhou/Reinforcement-learning-with-tensorflow (visited on 18th Nov. 2018).