agent are uniformly distributed. Given the random initialisation of task and agent locations and deadlines, it is sometimes impossible for some tasks to be started by any agent before its deadline.

The training set is generated by running task allocation experiments under various configurations. The task and agent numbers were selected to cover a range from under-constrained to over-constrained. Over-constrained signifies that there are a greater number of tasks than can be assigned given the time constraints, while under-constrained signifies that there is enough capacity to assign all tasks. Each observation was labeled corresponding to whether CBBA_{EDF} or CBBA_{NTF} yielded the highest number of allocated tasks overall at the time of convergence. Under a star communication network topology, the number of agents was fixed at: 14, and the numbers of tasks were: 84, 112, 140, 168, 196, 266. Under a fully connected communication network topology, the numbers of agents were: 4, 6, 8, 10, 12, 14, 16 and the number of tasks was fixed at: 130. To add variation, this latter setup was repeated with both agent types able to service both task types. The increase in number of tasks and agents were arbitrarily selected within a range to cover a variety of tasks to agent ratios, from under-constrained to over-constrained. Each setup was run 50 times with the same configuration but different initial conditions.

From simulations running these configurations with CBBA_{EDF} and CBBA_{NTF}, the observations: $[|\mathbf{o}|, |\bar{\mathbf{a}}|]/\ m_c$, were taken from each agent at each iteration starting from $T = 2$ to the time of convergence. Given the high number of agents deployed in one scenario and the repetition of scenarios, input vectors $[|\mathbf{o}|, |\bar{\mathbf{a}}|]/\ m_c$ with identical values and labels may be observed in the dataset. Such data points are effectively duplicates and can be safely removed from the dataset. After removal of duplicates, the labeled data set consisted of approximately 6000 unique observations. Cases for which the two heuristics were equivalent were left in.

4 PERFORMANCE ANALYSIS

The simulation results compare the performances of the different algorithms with respect to average task allocations and iterations until convergence at the end of the task allocation process. In real-time systems, the time to reach a solution may be critical to successfully completing the mission. Thus, our analysis also investigates whether strategy adaptation allows the system to converge to a solution within similar time to non-adaptive algorithms. The total iterations for one simulation is determined by the last time an allocation change was made by any agent, either through inclusion or removal. A marginal increase in average execution time per iteration is expected with the adaptive strategies compared to the non-adaptive algorithms. In real-time settings, variable factors that depend on the specific implementation, such as the time required for communication, the processing speed, the number of tasks, the number of times the agent attempts to make a prediction, are all factors that may impact the proportional increase in average execution time. These points are worth investigating in future work to evaluate the trade-off. Results are shown as averages over 50 runs.

Table 1: Scenario Specification

	Medicine	Food
Agent Speed	30m/s	50m/s
Agent Battery	Between 2500 and 5000 seconds	
Agent Start Position	10 000m x 10 000m x 0m ground space	
Task Duration	300 seconds	350 seconds
Task Deadline	Between 0 and 5000 seconds	
Task Location	10 000m x 10 000m x 1000m 3D space	

4.1 Unseen Row Topology, Task Numbers, and Rank Conflict Resolution

This section shows the results of tests comparing the algorithms operating with 14 agents under conditions not seen in training: under a row topology, with different task numbers, and a different conflict resolution strategy. In Figure 2(a), the proposed $\text{CBBA}^+{}_{\text{NN}}$ and $\text{CBBA}^+{}_{\text{SVM}}$ both match the best average numbers of allocations achieved by the non-adaptive approaches showing that the agents are correctly predicting and selecting the optimal heuristic under different conditions. The number of iterations until convergence are similar for the proposed adaptive approach and the non-adaptive approach that the agents are selecting, indicating that strategy adaptation maintains a similarly low number of iterations as the non-adaptive cases. $\text{CBBA}^+{}_{\text{NN}}$ takes marginally longer to converge on average than CBBA_{EDF} and $\text{CBBA}^+{}_{\text{SVM}}$ at 130 tasks, but matches the fastest convergence time of CBBA_{NTF} at 220 and 250 tasks. $\text{CBBA}^+{}_{\text{SVM}}$ is relatively faster to converge for the lower task numbers and relatively slower for the higher task numbers compared with $\text{CBBA}^+{}_{\text{NN}}$.

Figure 2(b) shows the results with all algorithms using the Rank-based conflict resolution strategy, where agents resolve conflicts on task assignments based on agents' ranks. $\text{CBBA}^+{}_{\text{NN}}$-Rank still matches the best allocation numbers compared with the non-adaptive approaches, mostly unaffected that Rank consensus was not seen during training. $\text{CBBA}^+{}_{\text{SVM}}$-Rank matches the best average numbers of allocations for the lower task numbers, but for the higher numbers shows a drop in performance compared with $\text{CBBA}^+{}_{\text{NN}}$-Rank. However, $\text{CBBA}^+{}_{\text{SVM}}$-Rank still allocates significantly more tasks on average than CBBA_{EDF}-Rank. The time to convergence for each algorithm is faster overall with Rank consensus, and average time taken is comparable for each algorithm. $\text{CBBA}^+{}_{\text{NN}}$-Rank takes at most 2 extra iterations on average than the slowest non-adaptive algorithm, and at best 1 iteration less. $\text{CBBA}^+{}_{\text{SVM}}$-Rank is the slowest to converge.

Figure 2(c) shows that when agents use the NTF heuristic in the scenarios with the higher numbers of tasks, the agents allocate more tasks overall on average if combined with Rank conflict resolution. We repeated the experiments for $\text{CBBA}^+{}_{\text{NN}}$ with the added condition that if an agent predicts that NTF is the optimal heuristic, it also switches to using Rank conflict resolution. The results are plotted as NN-Switch. Figure 2(c) shows that for the higher number of tasks, NN-Switch benefits from the higher allocations enabled by Rank conflict resolution, as well as the faster convergence time compared with $\text{CBBA}^+{}_{\text{NN}}$ and CBBA_{NTF}. For the higher number of tasks, the convergence time for NN-Switch is closest to $\text{CBBA}^+{}_{\text{NN}}$-Rank, which has the fastest convergence. In the lower task numbers, NN-Switch benefits from the higher allocations afforded by using bids for conflict resolution, and matches the

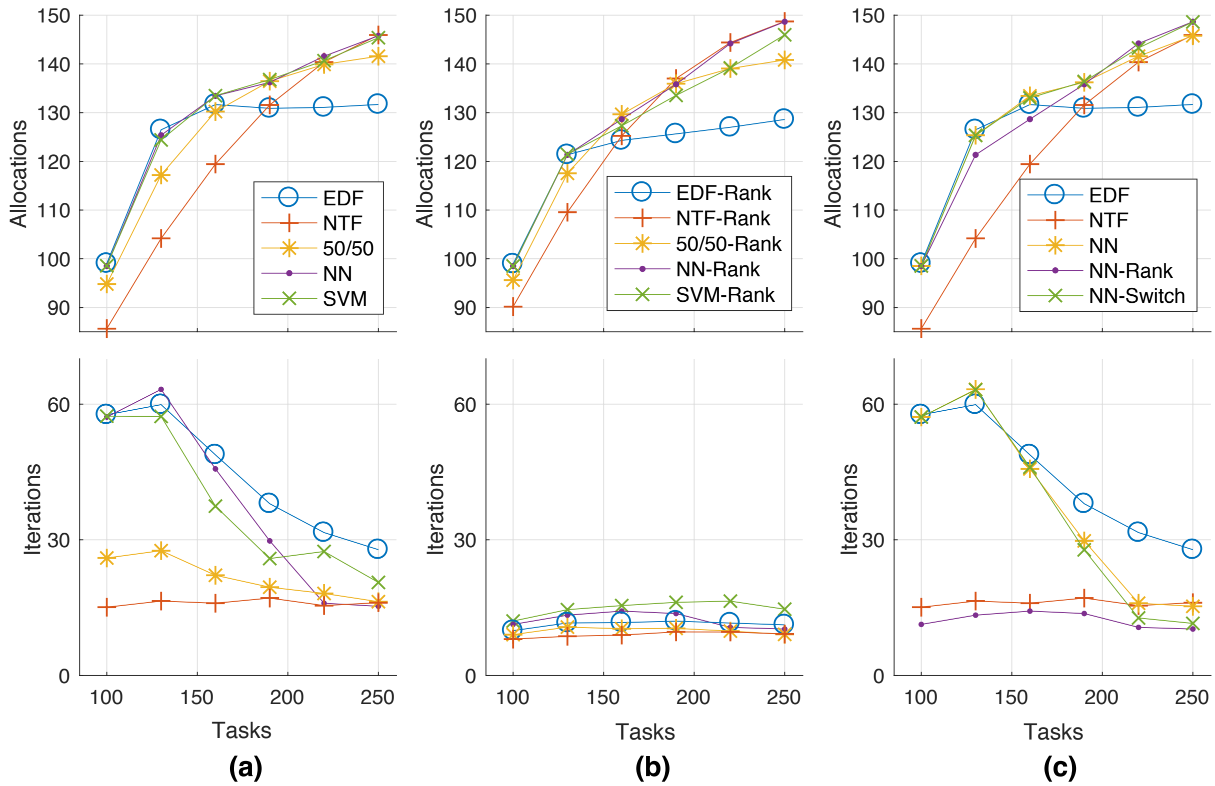

Figure 2: Average task allocations (top) and average iterations until consensus (bottom) for scenarios with different task numbers (100,130,160,190,220,250) and a fixed number of networked agents (14), connected with a row topology. In a) five algorithms are compared: all agents self-assign tasks with the earliest-deadline-first (EDF) heuristic; all agents self-assign tasks with the nearest-task-first (NTF) heuristic, agents are split half and half into using EDF and NTF respectively (50/50); agents are initialised with 50/50 and then optionally switch to EDF or NTF based on a trained neural network (NN) prediction; agents are initialised with 50/50 and then optionally switch to EDF or NTF based on a support vector machine (SVM) prediction. In b) the same algorithms resolve conflicts according to the relative ranking of agents (Rank). In c) with NN-Switch, the task inclusion is as with NN, and conflict resolution switches to Rank if the optimal task inclusion heuristic is predicted to be NTF.

slower convergence times of $CBBA^{+}_{NN}$ and $CBBA^{+}_{EDF}$. In these scenarios, the task inclusion strategy and the consensus strategy both affect the performance of the task allocation algorithm.

4.2 Unseen Agent Numbers and Task Numbers

This section shows the results of tests comparing the algorithms operating with different and varying numbers of agents, as well as task numbers unseen in training. Figure 3(a) and Figure 3(b) plot results with a fixed number of agents (10) and different numbers of tasks unseen in training. In Figure 3(a) the topology is fully connected and in Figure 3(b) it is star connected. Under both the full and star topologies, $CBBA^{+}_{SVM}$ is able to consistently match the average allocations achieved by the best non-adaptive approaches in a comparable convergence time. $CBBA^{+}_{NN}$ performs marginally less well in this scenario compared with $CBBA^{+}_{SVM}$. With the fully connected topology, $CBBA^{+}_{NN}$ falls short of achieving the highest average allocations for 4 out of the 6 task numbers. With the star topology, $CBBA^{+}_{NN}$ only falls short once when the best non-adaptive algorithm is $CBBA_{50/50}$, which is not used for training. The convergence times of the proposed adaptive approaches are again comparable to the non-adaptive baseline approaches.

Figure 3(c) plots results for simulations with different unseen agent numbers and a fixed number of tasks (130) under a fully connected network. With the higher number of agents (11,13, and 15), $CBBA^{+}_{NN}$ and $CBBA^{+}_{SVM}$ perform well in allocating tasks by accurately predicting and selecting the optimal heuristic. The proposed adaptive algorithms perform less well for the lower number of agents (5 and 7). $CBBA^{+}_{SVM}$ matches $CBBA_{50/50}$ for number of allocations which achieves the second highest average allocations of the non-adaptive approaches, while $CBBA^{+}_{NN}$ predicts incorrectly that EDF is the optimal heuristic. With 5 and 7 agents, $CBBA^{+}_{NN}$ and $CBBA^{+}_{SVM}$ also converge marginally slower than the non-adaptive approaches.

It is worth noting that $CBBA_{50/50}$ performs well in all the tested scenarios and offers good convergence speed. Adjusting the ratio to have more agents using EDF proportionally increases the average number of allocations for the lower task numbers, and reduces the average number of allocations for the higher task numbers. The

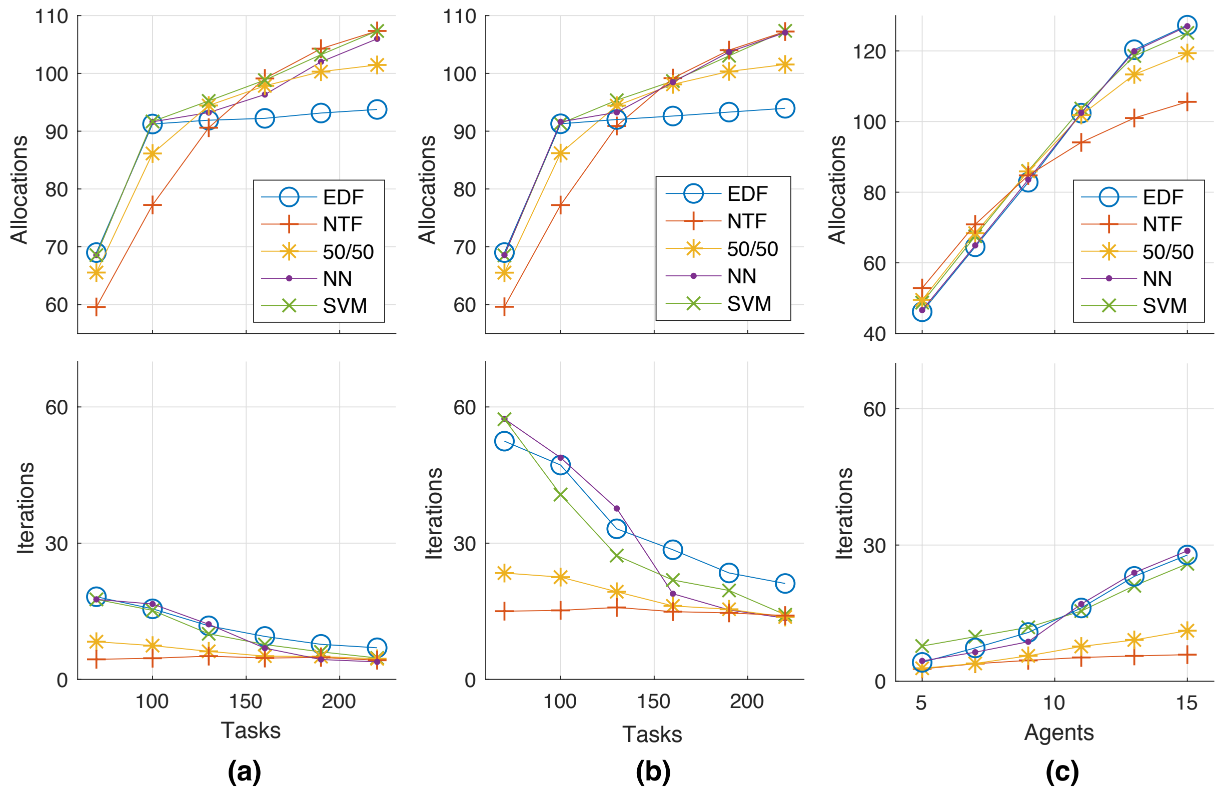

Figure 3: Average task allocations (top) and average iterations until consensus (bottom). In a) and b) task numbers are (70,100,130,160,190,220) with a fixed number of agents (10). In a) agents are connected with a fully connected topology, in b) a star topology. In c) a fixed number of tasks (130), agent numbers are (5,7,9,11,13,15) with a fully connected topology. The algorithms using EDF, NTF, 50/50, NN, and SVM are compared.

inverse holds true when the ratio favours agents using NTF. For simple problems, this static approach is a viable alternative to the proposed approach. The proposed adaptive approach instead offers a proof of concept that can be extended for more complex scenarios. In fact, more sophisticated heuristics can be added or learned to give agents greater adaptability and ability to optimise the task allocation. Such an increase in flexibility and ability to optimise the task allocation would justify the use of the proposed adaptive approach compared to a static approach.

5 CONCLUSIONS

This study investigated the possibility and potential performance gain of enabling distributed agents to independently adapt their task allocation strategies according to locally received information. An adaptive distributed approach is proposed that combines a prediction function with a decision making capability to select the predicted optimal strategy. Results showed that in the majority of scenarios tested, a performance gain was achieved by using the proposed approach. Agents were able to predict and select the optimal task inclusion heuristic to optimise the number of allocated tasks. In a minority of cases tested, when the number of agents was lowest, the agents predicted the incorrect heuristic. However, this resulted in a performance no worse than the non-adaptive strategy. Preliminary results showed that agents could further optimise the task allocation by adapting their conflict resolution strategy.

Factors such as the training data, the inputs, the machine learning tool, and the time of the prediction, are all factors that may impact the accuracy of the predictions, and are therefore interesting points to consider more deeply in future work. The proposed method could be extended to support a greater number of heuristics. For problems of greater complexity, additional inputs could be tested for increased accuracy. Additional inputs may include the number of tasks an agent removes during a round due to conflicts. Furthermore, the proposed approach could be adapted to support agents in learning the best strategy online, as well as adapting to changing optimisation objectives.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions.

REFERENCES

[1] Giulio Binetti, David Naso, and Biagio Turchiano. 2013. Decentralized task allocation for surveillance systems with critical tasks. *Robotics and Autonomous Systems* 61, 12 (2013), 1653–1664. https://doi.org/10.1016/j.robot.2013.06.007

[2] Lucian Busoniu, Robert Babuska, and Bart De Schutter. 2008. A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 2 (2008), 156–172. https://doi.org/10.1109/TSMCC.2007.913919

[3] Han-Lim Choi, Luc Brunet, and Jonathan P. How. 2009. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics* 25, 4 (2009), 912–926.

[4] Han-Lim Choi, Andrew K. Whitten, and Jonathan P. How. 2010. Decentralized task allocation for heterogeneous teams with cooperation constraints. In *American Control Conference (ACC)*. IEEE, 3057–3062.

[5] Rongxin Cui, Ji Guo, and Bo Gao. 2013. Game theory-based negotiation for multiple robots task allocation. *Robotica* 31, 6 (2013), 923–934. https://doi.org/10.1017/S0263574713000192

[6] Donato Di Paola, Andrea Gasparri, David Naso, and Frank L. Lewis. 2014. Decentralized dynamic task planning for heterogeneous robotic networks. *Autonomous Robots* 38 (2014), 31–48. https://doi.org/10.1007/s10514-014-9395-y

[7] M. Bernardine Dias, R. Zlot, N. Kalra, and A. Stentz. 2006. Market-Based Multirobot Coordination: A Survey and Analysis. *Proc. IEEE* 94, 7 (2006), 1257–1270. https://doi.org/10.1109/JPROC.2006.876939

[8] Andrew Garland and Richard Alterman. 2004. Autonomous agents that learn to better coordinate. *Autonomous Agents and Multi-Agent Systems* 8, 3 (2004), 267–301.

[9] Brian P. Gerkey and Maja J. Matarić. 2004. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research* 23, 9 (2004), 939–954. https://doi.org/10.1177/0278364904045564

[10] Matthew Gombolay, Reed Jensen, Jessica Stigile, Sung-Hyun Son, and Julie Shah. 2016. Apprenticeship Scheduling: Learning to Schedule from Human Experts. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 826–833. http://dl.acm.org/citation.cfm?id=3060621.3060736

[11] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics* 4, 2 (1968), 100–107.

[12] Yujing Hu, Yang Gao, and Bo An. 2015. Learning in Multi-agent Systems with Sparse Interactions by Knowledge Transfer and Game Abstraction. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*. IFAAMAS, 753–761.

[13] Yichuan Jiang. 2016. A survey of task allocation and load balancing in distributed systems. *IEEE Transactions on Parallel and Distributed Systems* 27, 2 (2016), 585–599.

[14] Luke Johnson, Sameera Ponda, Han-Lim Choi, and Jonathan P. How. 2010. Improving the Efficiency of a Decentralized Tasking Algorithm for UAV Teams with Asynchronous Communications. In *AIAA Guidance, Navigation, and Control Conference*. 8421.

[15] Luke B. Johnson, Han-Lim Choi, and Jonathan P. How. 2016. The Role of Information Assumptions in Decentralized Task Allocation: A Tutorial. *IEEE Control Systems* 36, 4 (2016), 45–58.

[16] Luke B. Johnson, Han-Lim Choi, Sameera Ponda, and Jonathan P. How. 2012. Allowing Non-Submodular Score Functions in Distributed Task Allocation. In *51st IEEE Conference on Decision and Control (CDC)*. 4702–4708.

[17] G. Ayorkor Korsah, Anthony Stentz, and M. Bernardine Dias. 2013. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research* 32, 12 (oct 2013), 1495–1512. https://doi.org/10.1177/0278364913496484

[18] Michael L. Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the eleventh international conference on machine learning (ICML)*. 157–163.

[19] Hakim Mitiche, Dalila Boughaci, and Maria Gini. 2015. Efficient Heuristics for a Time-Extended Multi-Robot Task Allocation Problem. In *First International Conference on New Technologies of Information and Communication (NTIC)*. IEEE, 1–6.

[20] Alexandru Iulian Orhean, Florin Pop, and Ioan Raicu. 2017. New scheduling approach using reinforcement learning for heterogeneous distributed systems. *J. Parallel and Distrib. Comput.* (2017). https://doi.org/10.1016/j.jpdc.2017.05.001

[21] Liviu Panait and Sean Luke. 2005. Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems* 11, 3 (2005), 387–434.

[22] Sameera Ponda, Josh Redding, Han-Lim Choi, Jonathan P. How, Matt Vavrina, and John Vian. 2010. Decentralized planning for complex missions with dynamic communication constraints. In *American Control Conference (ACC)*. IEEE, 3998–4003.

[23] Sameera S. Ponda. 2012. *Robust Distributed Planning Strategies for Autonomous Multi-Agent Teams*. Ph.D. Dissertation.

[24] Sameera S. Ponda, Luke B. Johnson, Andrew N. Kopeikin, Han-Lim Choi, and Jonathan P. How. 2012. Distributed planning strategies to ensure network connectivity for dynamic heterogeneous teams. *IEEE Journal on Selected Areas in Communications* 30, 5 (2012), 861–869. https://doi.org/10.1109/JSAC.2012.120603

[25] Peter Stone and Manuela Veloso. 2000. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots* 8, 3 (2000), 345–383.

[26] Milind Tambe. 1997. Towards flexible teamwork. *Journal of artificial intelligence research* 7 (1997), 83–124.

[27] Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*. 330–337.

[28] Joanna Turner, Qinggang Meng, Gerald Schaefer, and Andrea Soltoggio. 2018. Fast Consensus for Fully Distributed Multi-Agent Task Allocation. In *The 33rd Symposium On Applied Computing (SAC '18)*. ACM/SIGAPP. https://doi.org/10.1145/3167132.3167224

[29] Joanna Turner, Qinggang Meng, Gerald Schaefer, Amanda Whitbrook, and Andrea Soltoggio. 2017. Distributed Task Rescheduling With Time Constraints for the Optimization of Total Task Allocations in a Multirobot System. *IEEE Transactions on Cybernetics* 99 (2017), 1–15.

[30] Gerhard Weiß. 1995. Adaptation and learning in multi-agent systems: Some remarks and a bibliography. In *International Joint Conference on Artificial Intelligence*. Springer, 1–21.