COMP34111 - Journal Report

Submission Information

- **Course:** COMP34111
- **Academic Year:** 2025–26
- **Student Name:**
  Benjamin H Kidd
- **Student ID:**
  11019777
- **Group:**
  24
- **Deadline:** 19th December, 2pm
- **Note:** Each student must submit their own copy. No anonymous submissions.

1 Individual Contribution and Reflection (20%)

1.1 Summary of My Contributions

In this section, describe clearly and honestly what **you** have done for the project so far. Mention specific tasks, responsibilities, and any deliverables you produced.

1.2 Reflection on Group Work

- None

1.3 Personal Development

Comment on:

- None

2 Group Meeting Log (30%)

Log

Meeting #1

Date: 21/11/2025
Time: 10:00–12:30
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None

Tasks Agreed :

Task ID	Responsible	Due Date	Description
T1	Ben	12/12	Utilize an Alpha Zero Approach, using MCTS & Deep Learning.
T2	Daniel	09/12	Evalutation function.
T3	Daniel	12/12	Alpha Beta.
T4	Naddy	12/12	Manual MCTS (Policy Selection, etc).
T5	David	12/12	Explore evolutionary stategies for Deep learning

Rationale for Decisions

- Separating individual approaches allowed for the most parallelisation of work.
- We felt that within the unit, the main approaches available were AlphaBeta (MiniMax), MCTS and AlphaGo (AlphaZero), which could be expanded on using other approaches including the evaluation and evolution strategies. We decided to attempt all of these, and then evaluate which performed the best at the end.

Meeting #2

Date: 25/11/2025
Time: 13:00–15:00
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None

Tasks Agreed None (Collectively wrote and submitted the group proforma).

Meeting #3

Date: 28/11/2025
Time: 10:00–13:00
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- Daniel (11017362)

Apologies for Absence

- David (11054893), Another coursework submission (NLP).

Tasks Agreed :

Task ID	Responsible	Due Date	Description
T6	Naddy	30/11	Research relevant papers on how to implement MCTS in Hex.
T7	Daniel	02/12	Create a visualiser for the evaluation function.
T8	David	02/12	Research relevant papers and start work towards an implementation of a genetic strategy

Rationale for Decisions

- There are plenty of resources available from researchers who have optimised Hex agents before. Their findings will help with making the algorithm faster.
- It is difficult to identify if what the evaluation function is doing makes sense or not, so we should have an easy way to see what it thinks.
- There are several papers on implementing evolutionary strategies to teach neural networks to play Hex and other similar games. While this approach was not widely used, it did look promising as an alternative which may outperform more simple agents.

Meeting #4

Date: 02/12/2025
Time: 13:00–15:00
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None

Tasks Agreed :

Task ID	Responsible	Due Date	Description
T9	All	07/12	Create and fill out our presentation.
T10	Naddy	09/12	Run some experiments with the MCTS implementation.
T11	Daniel	09/12	Tune heuristics used in evaluation function. Determine if alpha beta is worthwhile spending time on.
T12	Ben	09/12	Implement prioritisation to imply smaller games are better.
T13	David	09/12	Implement simple neural agent and look into strategies to evolve it

Rationale for Decisions

- Our presentation is coming up soon and we want to get it done sooner rather than later, so it won't get in the way of making our models.
- The presentation requires some experimental results so ideally the vanilla MCTS implementation should be working and winning before then.
- The current heuristics were very simple, and we need a way to score patterns. Even so, alpha beta might not be promising enough.
- The AlphaZero approach was winning consistently, but was taking almost full boards to win.
- The AI often takes full games when it could win in fewer moves, so we should prioritise winning faster. Thus I reward the AI more for winning in fewer moves.

Meeting #5

Date: 05/12/2026
Time: 10:00–12:00
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None

Tasks Agreed None (Collectively wrote the group powerpoint).

Meeting #6

Date: 08/12/2025
Time: 13:00–15:00
Location: Hybrid, Kilburn and Discord

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None

Tasks Agreed :

Task ID	Responsible	Due Date	Description
T14	All	09/12	Script and rehearse the presentation.
T15	Naddy	11/12	Look into implementing RAVE or other optimisations for MCTS.
T16	Daniel	11/12	Create bitmasks for common patterns to speed up recognition of said patterns.
T17	Ben	15/12	Implement a minimax checker for AlphaZero to help with game endings.

Rationale for Decisions

- We need to ensure we're within the time limit of the presentation to get everyone's input.
- MCTS was not quick enough at finding optimal moves on a board as big as 11x11.
- Evaluation was slow when ran with alpha beta minimax, bit manipulation operations would speed it up significantly.
- Game endings were struggling to be found, sometimes with the AI having a winning board state but never playing it. Minimax used at the start of the move to check up to a given depth (which increases based on the number of moves played to keep compute power reasonable), playing winning moves if found, or blocking moves if opponent winning moves found. If no certain win/loss moves are found, normal AlphaZero is used.

Meeting #7

Date: 09/12/2025
Time: 13:00–15:00
Location: Room (Tootil)

Members Present

- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- Ben (11019777), CGI Assessment Centre.

Tasks Agreed :

Task ID	**Responsible**	**Due Date**	**Description**
T18	Daniel	11/12	Explore any other potential approaches.
T19	Naddy	14/12	Run optimised MCTS implementation against other models.
T20	Naddy	14/12	Replace random rollouts with heuristic-based rollouts.
T21	Ben	15/12	Utilise Supervised Learning (AlphaGo style) for initial learning.

Rationale for Decisions

- Minimax approach not promising, explore any other potential approaches we havent considered.
- This will be done to find the MCTS implementation's weaknesses, and to give guidance on what to focus a heuristic on.
- While the model was learning well, it would take lots of games to get strong. Utilise known good games to train the AI initially, hoping it will pick up on some good initial patterns/weights.

Meeting #8

Date: 12/12/2025
Time: 10:00–13:00
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None.

Tasks Agreed :

Task ID	Responsible	Due Date	Description
T22	Naddy	16/12	Research further optimisations for the MCTS implementation and add them.
T23	Daniel	16/12	Apply h-search and minimax optimisations to MCTS approach.
T24	Ben	18/12	Utilise existing KataHex model since the coursework allows it, to avoid extensive training being required.

Rationale for Decisions

- The MCTS implementation is performing decently, but it still takes too long per turn to reach that point. May considering moving away from python.
- H search would help optimise MCTS expansion policy.
- While AlphaZero was learning correctly, it would need massive amounts of computing power to train to a decent level. Since the discussion board says we can use pre-trained models, I implement a bot based on the existing KataHex model.

Meeting #9

Date: 15/12/2025
Time: 13:00–15:00
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None.

Tasks Agreed :

Task ID	Responsible	Due Date	Description
T25	Ben (1101977)	18/12	Complete their journal version
T26	Naddy (11011468)	18/12	Complete their journal version
T27	David (11054893)	18/12	Complete their journal version
T28	Daniel (11017362)	18/12	Complete their journal version
T29	Ben	18/12	In the same way as with the AlphaZero approach, add Minimax for KataHex.

Rationale for Decisions

- T29. Same rationale as T17 to help with endgames.
- Decision made to submit KataHex bot. The best bots were AlphaZero, MCTS and KataHex. MCTS beat AlphaZero most of the time, but KataHex beat both all the time.

Meeting #10

Date: 18/12/2025
Time: 12:00–14:00
Location: Room (Tootil)

Members Present

- Ben (11019777)
- Naddy (11011468)
- David (11054893)
- Daniel (11017362)

Apologies for Absence

- None.

Tasks Agreed None (Collectively finalised project).

2.1 Compact Summary of Meetings (Optional)

Table 1: Summary of Group Meetings

Date & Time	Location	Members Present	Main Outcomes / Tasks
21/11/2025, 10:00	Room	Ben, Naddy, David, Daniel	Defined Individual Approaches.
25/11/2025, 13:00	Room	Ben, Naddy, David, Daniel	Started work on current approaches, discussed methods of implementation.
28/11/2025, 10:00	Room	Ben, Naddy, David, Daniel	Submitted Proforma and continued work on current approaches, updated teammates on progress.
2/12/2025, 13:00	Room	Ben, Naddy, Daniel	Continued work on current approaches, updated teammates on progress. Considering removing Alpha Beta from scope.
05/12/2025, 10:00	Room	Ben, Naddy, David, Daniel	Created Powerpoint.
08/12/2025, 13:00	Hybrid	Ben, Naddy, David, Daniel	Finalised and Practiced Powerpoint

Date & Time	Location	Members Present	Main Outcomes / Tasks
09/12/2025, 13:00	Room	Naddy, David, Daniel	Continued work on current approaches, updated teammates on progress. Decided to remove Alpha Beta from scope.
12/12/2025, 10:00	Room	Ben, Naddy, David, Daniel	Continued work on current approaches, updated teammates on progress. Started work on KataHex approach.
15/12/2025, 13:00	Hybrid	Ben, Naddy, David, Daniel	Compared bots and finalised log. Added final MiniMax improvements to KataHex agent.
18/12/2025, 12:00	Hybrid	Ben, Naddy, David, Daniel	Finalised everything for submission

3 Bot Method and Unique Strengths (50%)

3.1 Method Overview

Describe what method your bot used. For example:

- Algorithm(s) or model(s) used
- High-level architecture
- How the components interact

We originally implemented four different approaches to the problem: vanilla MCTS, an AlphaZero-based approach, alphabeta minimax, and evolutionary strategies for training a simple neural network. After testing, we found that the AlphaZero and MCTS approachs were the strongest, so we focused on improving them further. The other approaches all had various issues, such as being too slow (minimax) or not strong enough (evolutionary strategies). MCTS was typically better than AlphaZero, winning around 60-70% of games in head-to-head matches. This was likely due to the limited training time available for AlphaZero, even after a week of training, as this is still only thousands of games not millions like AlphaGo.

As such, and due to the allowance of it, we decided to use an existing pre-trained model from KataHex instead for our final submission. The KataHex model beat MCTS and AlphaZero consistently in testing, so we focused on improving it further by adding a minimax endgame solver to help it find winning sequences of moves to finsh games, something base KataHex struggled with. KataHex uses a similar approach to AlphaZero, using a combination of MCTS and a neural network to evaluate the game state and select moves. The minimax addition helps it to find winning sequences of moves in the endgame more reliably. It searches until a certain depth for solvable game states, and plays winning moves if found, or blocks opponent winning moves if found. If no certain win/loss moves are found, normal KataHex is used. The depth it searches to scales with the number of moves played in order to keep compute time reasonable.

3.2 Technical Details

Provide more detail here, such as:

- Input representation and features
- Training, tuning, or rule design
- Any libraries, frameworks, or tools used
- Evaluation strategy and results

The agent runs the `hex3_27x_b28.bin.gz` model, a model trained on roughly 6 RTX4090 for one month, which has millions of games of experience. It works in a similar way to our AlphaZero style agent, utilising a nerual network of CNN layers.

The agent handles the creation of the KataHex process from its executable, and uses Hex GTP for communicaiton with it. GTP (Go Text Protocol) was originally used for KataGo, and has been re-purposed for use in hex. This language is used to setup the board, send moves played and ask for the next best (predicted) move. This then gets played. The board is completely synchronised for the first 3 moves to account for possible colour changes. After that, the board is only fully re-synchronised with the KataHex process if the process dies, in order to save on compute.

Significant tuning was performed on `config.cfg` to adapt the engine for the CPU-constrained Docker environment. Most of the performance settings were kept the same, such as exploration. Hardware limits were enforced by enforcing limits in this configuration. There was significant experimentation and benchmarking done during this time to ensure adequate performance and reasonable time.

KataHex was very good at getting into winning states, but often struggled to actually win. Thus the game solver (alpha beta style) was used, to explore the game tree to certain depths, and actually finish the games once solved. the solver would only return moves which would certainly lead to a win, regardless of what the opponent played. The model is used if this was not found.

For the solver, to balance accuracy with speed, the HexSolver calculates search depth dynamically using a log formula based on the number of moves completed. This allows the solver to search deeper in the endgame (up to full board solution) while remaining shallow in the opening.

3.3 Unique Aspects and Strengths

Explain what made your bot *strong and powerful*:

- Novel ideas or design decisions
- Strengths compared to a baseline or naive approach
- Robustness, efficiency, scalability, etc.

Unlike approaches requiring extensive training from scratch, our implementation leverages KataHex's neural network pre-trained on millions of games. This model immediately provided strong positional understanding, allowing the bot to recognize winning patterns and strategic formations that would take conventional MCTS thousands of simulations to discover.

While deep neural networks excel at long-term strategic planning, they occasionally miss immediate wins, which could force winning sequences that end the game within a handful of moves. Our hybrid approach addresses this weakness by using a simple Minimax solver with depth calculations to identify guaranteed forced wins. The solver looks a given depth ahead to try and solve the game board, finding winning or losing states. It can then either play in these winning states or block these losing states. This combination ensures that we are leveraging the full strageic advantage that KataHex provides, while guaranteeing that it does not lose in otherwise obvious scenarios, especially in the endgame.

To optimise runtime and to improve model speed, we configured KataHex to run efficiently using the CPU constraned docker container. By tuning parameters such as cache size, number of threads to utilise, we ensured that the most of the available resources were used, maximising the number of simulations we could run per turn within the strict time limit.

To ensure that longer runs would be able to be completed, we implemented a fail-safe mechanism that would ensure that the model would keep running, and restarting it if it had crashed. This was achieved by rerunning the model using the previous board states to recreate the current board state, allowing it to continue from where it has left off.

This model won in every game tested against Naive, MCTS, AlphaZero models, proving its effectiveness.

3.4 Limitations and Possible Improvements

Briefly acknowledge:

- Known weaknesses or failure cases
- Improvements you would make with more time

The fail-safe synchronization logic introduces slight latency during complex board states. In extremely tight time controls, the overhead of re-launching the process if it crashes could risk a timeout.

The solver is currently inefficient checking every avaliable nodes. It would be good to build possible chains, and only consider moves that could block or finish them to make it more efficient, allowing you to check more depth due to less computation being required as less nodes are checked. This could be done by only checking neighbouring nodes, instead of all nodes, and that is done recursively.

The bot currently relies on the neural network for opening moves (Turn 1–2). Implementing a hard-coded opening book for the "Swap" phase could save time and ensure optimal play against non-standard openings.

3.5 Group Member Signatures

Each member of the group should sign this section with their name and student ID.

Name	**Student ID**	**Signature / Confirmation**
Ben	11019777	I confirm the above description of the method and my contribution.
Naddy	11011468	I confirm the above description of the method and my contribution.
David	11054893	I confirm the above description of the method and my contribution.
Daniel	11017362	I confirm the above description of the method and my contribution.

A Further Rules (For Reference Only)

1. No anonymous submissions.
2. Every group member should attend every meeting. If you cannot attend (illness, job interview, etc.) let your group teammates know as soon as possible.
3. If your group is not functioning or someone is not engaging, responding to communications or attending meetings, please let Mingfei Sun (mingfei.sun@manchester.ac.uk) know as soon as possible.