

TENSOR

a game of equilibrium

by Tony Alicea

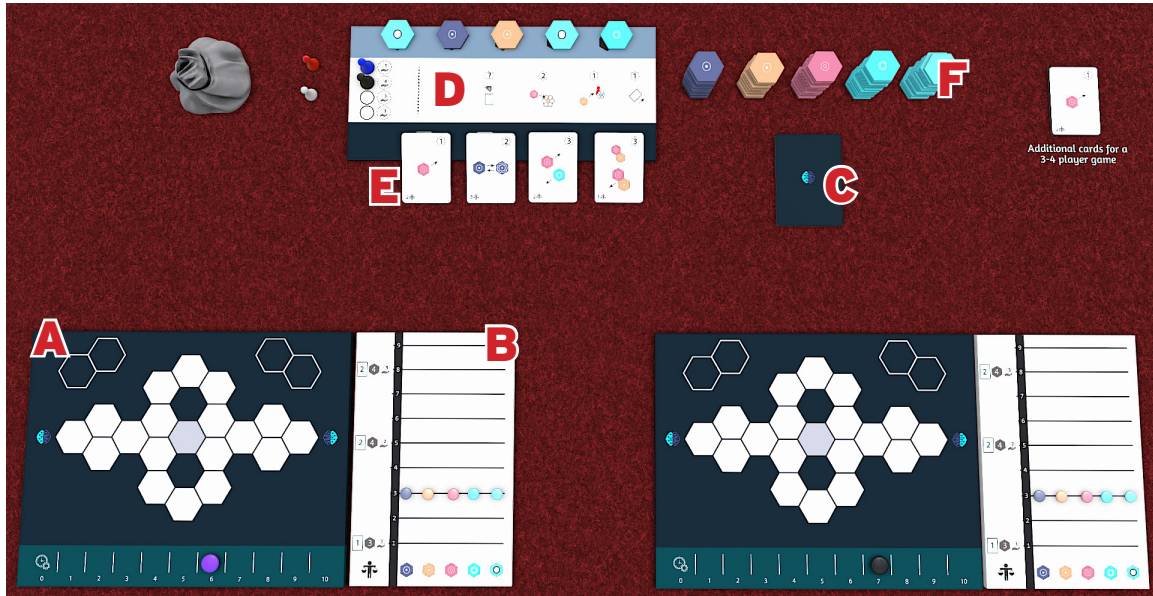
version 0.1



Tensor: A Game of Equilibrium



Welcome to Tensor! In this game players are managers of teams that are designing artificial intelligences. You are all working for the same research university, who will only give precious computer processing time to the teams whose AIs are doing the best. The game takes its inspiration from real-world “deep learning” techniques used when designing and building real AIs.

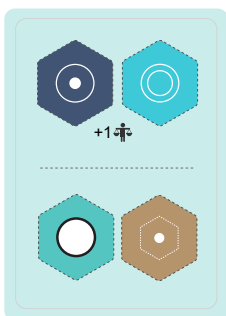
Unlike many games, Tensor is game of **equilibrium**, not efficiency. You will work to maintain your AI's neural network at a good equilibrium, and can hack other players AIs to throw their equilibrium off!



Setup

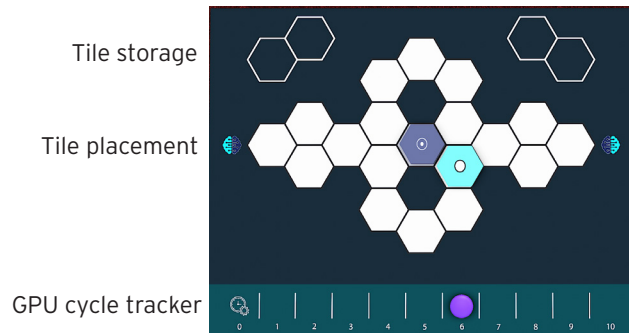
1. Each player chooses a color then takes a player board (A), a weight confidence board (B), and one Tensor card (C). Both the boards and the Tensor cards should be in a position where they can easily be seen by all players.
2. Place five weight tokens on line “3” of the confidence board (B) in the column of their corresponding color.
3. Place the action board (D) where all players can access it.
4. Place one of each type of Operation cards (E) on the action board. For a 3-4 player game, place two of each type of Operation cards instead.
5. Place the dataset tiles (F) on the table to form the **supply**.
6. From the supply, each player takes 3 tiles of each type (for a total of 15 tiles per player). Each player then examines their Tensor card, and places a tile in the center of their board that corresponds to one of the tiles in the top row of their Tensor card, and then, adjacent to that center tile, places a tile corresponding to one of the tiles in the bottom row of their Tensor card.

Example: Suppose below is the Tensor card a player drew. Then they would have the choice of  or  for their center tile, and one of the bottom row types for an adjacent tile. Thus, at the beginning of the game their board might appear as below (or another valid combination from their Tensor card).



Top row

Bottom row



Possible starting board

7. Each player now has 13 tiles. Gather all remaining tiles from all players together, shuffle, and place face down or in a bag to form the **draw pile**.
8. Decide which player will go first. Each player then places the tracker of their corresponding color on position "7" in the "GPU cycles" tracking area at the bottom of their player board. **The player who will go first instead places their tracker on position "6"**.
9. Place each player's turn order token on the action board, in the order they will be playing the first round (the starting player takes the top position, and, for the first round, play continues in clockwise order around the table).
10. Draw 5 tiles from the draw pile and place them on the action board.



Round Structure

Each round consists of many turns. On their turn, a player will take **one** and only one action of their choice (see the 5 actions listed in this rulebook). Players continue taking actions in turn order until all players pass. After all players pass, all players then simultaneously perform "Backpropagation" (see BACKPROPAGATION).

Players who pass first will go earlier in the turn order during the next round.



Action Costs

All actions cost GPU cycles, represented by the gear symbol. When a player spends or receives GPU cycles, the player updates the value on the GPU cycle tracker on their player board. While a player may pass at any time, if on their turn they cannot pay for any action then they must pass.



Thematic Explanation: Programming teams the players are managing all work for the same university, and are sharing the same clusters of computers to run their AIs. Real-world AIs are often trained using GPUs, or Graphical Processing Units, that were typically used for rendering high-end computer graphics. The university will give greater amounts of precious GPU processing time to well-performing AI systems.



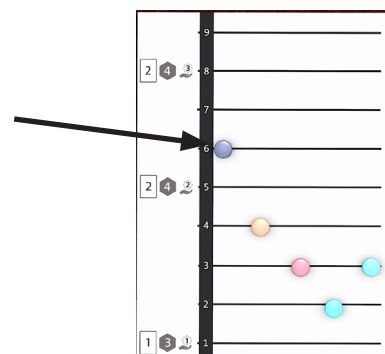
Hand and Storage Limits

The highest weight token on a player's confidence board dictates that player's hand and storage limits. The player will have a limit on the number of Operation cards they may have in their hand, and the number of dataset tiles they may have on the storage space of their player board. The position of the highest weight token on their confidence board also dictates how many extra GPU cycles they will receive as income at the end of the round.

Thematic Explanation: Higher performing programming teams are also rewarded by the university with more hard drive space for storage and complex operations.



Example: If a player's highest weight token (of any color) is **at or above** this symbol, then that means they may keep up to two Operation cards in their hand, up to four dataset tiles in storage on their board, and will receive two GPU cycles at the end of the round.





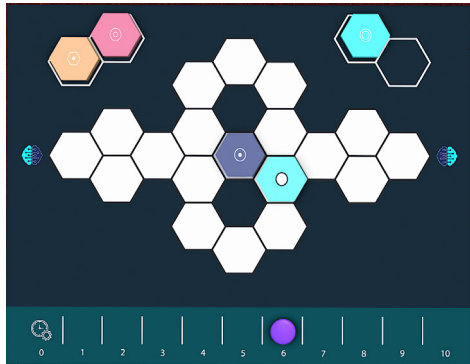
Action 1: Download Data

This action costs 2 GPU cycles. Fill your tile storage on your player board up to your limit by drawing tiles from the action board.

For example, if you have a tile storage limit of 3, and you already have one tile in storage on your board, then you must draw two tiles. If you had no tiles in storage on your board, then you would have to draw 3 tiles.

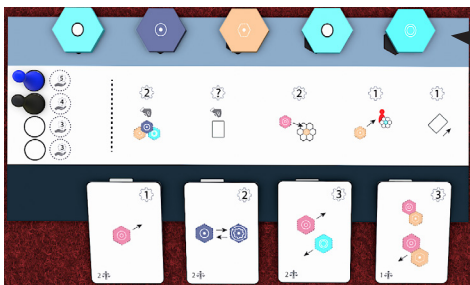
After taking tiles, replace them on the action board with tiles from the draw pile. Draw the number of tiles you took, and place the newly drawn tiles on the action board.

Thematic Explanation: To train an AI you need lots of data. Thousands if not millions of examples of what you want the AI to learn. This means a lot of downloads and a lot of storage. For example if you want to design an AI that can recognize a picture of a car, you need many example pictures of cars.



Example: This player has a maximum storage size of 3. Thus after taking the Download Data action, they have 3 tiles in their storage area at the top of their player board, ready to be used for other future actions.

NOTE: Tiles in storage stay in storage across rounds. They are only removed from storage via player actions.



At the end of the action, the tiles on the action board are replaced by drawing from the draw pile. Thus at the start of any player's turn, there will always be 5 tiles on the action board (except, possibly, during the game end).

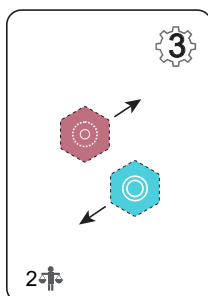


Action 2: Code a Tensor Operation

This action costs GPU cycles equal to the cost in upper right hand corner of the Operation card you draw. Draw **one** Operation card from the action board.

You may not carry out this action if it would take you over your hand limit as dictated by the confidence board.

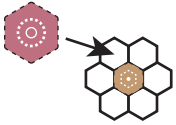
Thematic Explanation: Tensors are a way to structure and store data. Tensor operations are ways to then manipulate that data. These cards represent the act of coding, and then running the code, to carry out those operations on your data.



Example: This operation card costs 3 GPU cycles to draw into your hand.

IMPORTANT NOTE: Once you have a card in hand it is no longer available to other players. Thus it is possible for an Operation card of a particular type to be completely unavailable.

At the end of the round, however, all Operation cards in hand are returned to the action board. Thus you may draw a card to use it that round, or perhaps to keep someone else from using it!



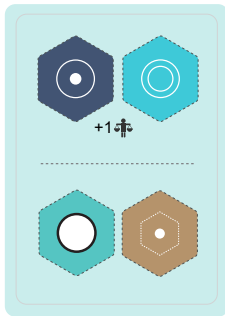
Action 3: Load Data

This action costs 2 GPU cycles. Take **one** tile from storage on your board and place it face-up on one of the light colored hexagons on your board.









Tile placement must conform to the following rules:


1. A tile must be placed adjacent to another tile.
2. Tiles whose types appear on your Tensor card must be placed adjacent to at least one corresponding tile of the adjacent type on that row of the Tensor card (see below example).

Thematic Explanation: This action represents loading downloaded data into your neural network, and conforming it to your chosen data structure.




Example: Suppose a player's Tensor card is the one shown here. For the Load Data action:

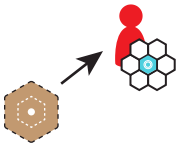
- a  tile *must* be placed adjacent to at least one  tile,
- a  tile *must* be placed adjacent to at least one  tile,
- a  tile *must* be placed adjacent to at least one  tile, and
- a  tile *must* be placed adjacent to at least one  tile.

Since a  is not on the Tensor card, it has no requirements other than it must be placed adjacent to another tile.

BONUS ACTION: Manual Tweak

If you place one of the two types of tiles that appear on the **top** line of your Tensor card, then you may immediately perform a manual tweak (+1 ) of your AI's neural network. You may move **one** weight token of your choice up **one** space on your confidence board.

This must be done immediately after placing the tile on your board. There is a reminder of this bonus action on the top line of each Tensor card.



Action 4: Ethical Hack

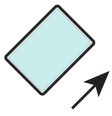
This action costs 1 GPU cycle. Take **one** tile from storage on your board and place it face-up on one of the light colored hexagons on **another player's** board.

Tile placement must conform to the 'Load Data' rules for that other player's board. Thus, the tile must be placed adjacent to another tile on that player's board and must follow the adjacency restrictions as dictated by **that other player's** Tensor card.

In other words, you perform a 'Load Data' action on that player's board but with a tile from your board's storage.

If you place a tile of a type that appears on the top line of that player's Tensor card, then that player may immediately perform a manual tweak of their confidence board, as specified under the rules of the Load Data action. This is a bonus action for that player, however, and play continues in normal turn order from the player who initiated the Ethical Hack.

Thematic Explanation: The university wants AI systems to be strengthened against hacking attempts. Thus the university encourages teams to attempt to hack each other's systems and provides some extra processing time to do so.

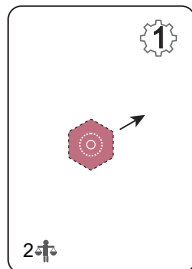


Action 5: Run a Tensor Operation

This action costs 1 GPU cycle (the card cost does not matter). Take **one** Operation card from your hand and return it to the action board. It is now available for other players to use. Then take the action according to the card, and carry out a number of manual tweaks, if possible, equal to what is found in the lower left of the card.

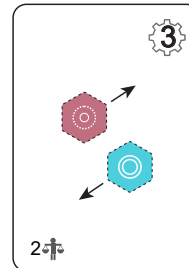
Operation card explanations are below.

Thematic Explanation: This action represents running the code that was written when the card was taken into your hand. Thus it allows you to manipulate your data in useful ways.



SUBTRACTION:

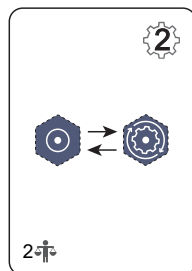
Remove any one tile from your board and place it in the supply.



ADDITION: Remove any one tile from your board and place it in the supply.

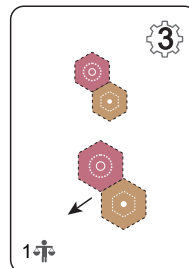
Then, take any one tile (of a different type than the one removed) from the supply and place it adjacent to any tile on your board.

Tensor card adjacency restrictions do **not** have to be followed, and you do **not** receive the Tensor card manual tweak bonus action.



DIVISION: Flip any face-down tile on your board over to face-up, or vice versa.

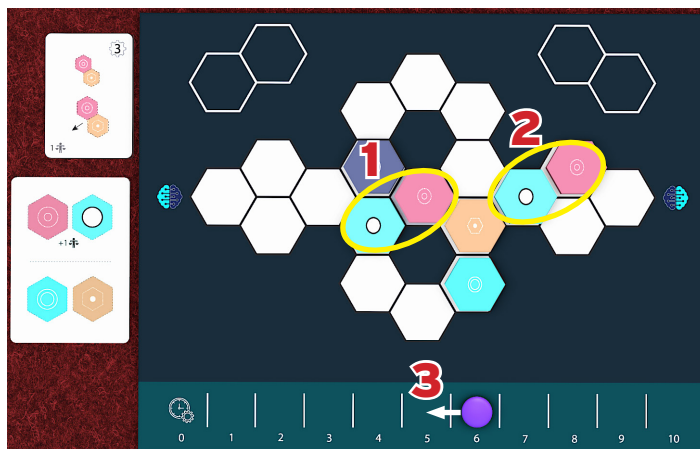
(For explanation of face-down tiles, see BACKPROPAGATION).



MULTIPLICATION: Choose any two adjacent tiles on your board (they may be of the same time) to duplicate. Take two matching tiles from the supply and place them on your board.

At least one of the two tiles being placed must be adjacent to a tile that is already on your board. The two tiles must be placed so that they are adjacent in the same orientation as the initially chosen tiles.

Tensor card adjacency restrictions do **not** have to be followed, and you do **not** receive the Tensor card manual tweak bonus action.



Example: The player is playing a multiplication Operation card. The card will be returned to the action board and they will pay 1 GPU cycle by moving the tracker as shown at position 3.

The player chose to duplicate the two tiles circles at position 1. They took matching tiles from the supply and placed them adjacent to another tile on their board at position 2.

Note that the two new tiles are placed in the same orientation as the originally chosen ones, duplicating their layout.

Note, also, that Tensor card adjacency restrictions were not followed. Even if they were, the player would not receive a Manual Tweak bonus action.

BONUS ACTION: Manual Tweak

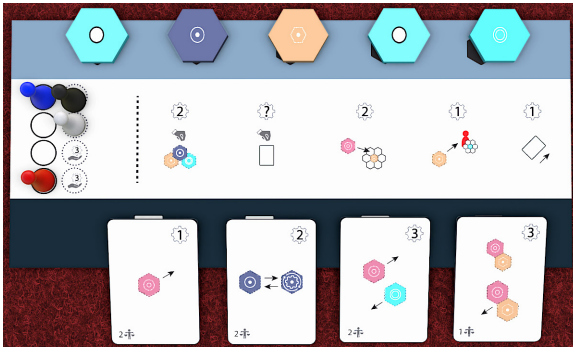
After playing an Operation card, you may immediately perform the number of manual tweaks (2 with a gear icon) of your AI's neural network as indicated in the bottom left of the card. You may apply all tweaks to the same weight token (thus moving it up multiple spaces), or split the tweaks amongst multiple tokens. Each tweak corresponds to moving one weight token of your choice up one space on the confidence board.

This must be done immediately after playing the Operation card.

NOTE: Good use of Operation cards, both for adjusting your board and manual tweaks, are keys to winning!

Action 6: Pass

On your turn you may choose to pass. Move your turn order token onto the topmost next available turn order spot on the action board, and immediately receive the indicated GPU cycles. You may no longer take any actions this round, and your subsequent turns are skipped. Once all players pass, the round ends.



Example: The player playing black has already passed, and it is the turn of the player playing white. They decide to pass.

They move their turn order marker to the next available space, which indicated that they immediately receive 4 GPU cycles.

Only the blue and red players will take further actions this round. Once all players pass, the round is over.



Round End

When the round ends, all players return any Operation cards in their hands to the action board. Then, slide the four turn order tokens on the action board to the left, thereby establishing the new turn order for the next round. When the next round begins, all turns will occur in that order.

Finally, each player receives GPU cycles according to their income, as dictated by the highest weight token on their confidence board. Then AI training is run! Carry out Backpropagation, as indicated below.

Round End: Backpropagation

Each player simultaneously examines their player board individually. A player counts the number of **unflipped (face-up)** tiles of each type on their board. They then move the corresponding weight token for each of the 3 types of tiles with the most number of face-up tiles up one space each on the confidence track.

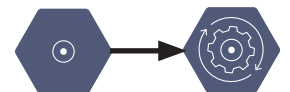
In other words, the top 3 (by unflipped quantity) tile types move **up** on the confidence board. The others move **down** one space on the confidence board.

If the number of types of unflipped tiles is less than 3, then do the same but for the top 2 or top 1 types of tiles. In this case, all types of tiles for which there are none unflipped will move down one space.

If there is a tie, then the tiebreaker is the order of types of tiles as they appear on the bottom of the confidence board.

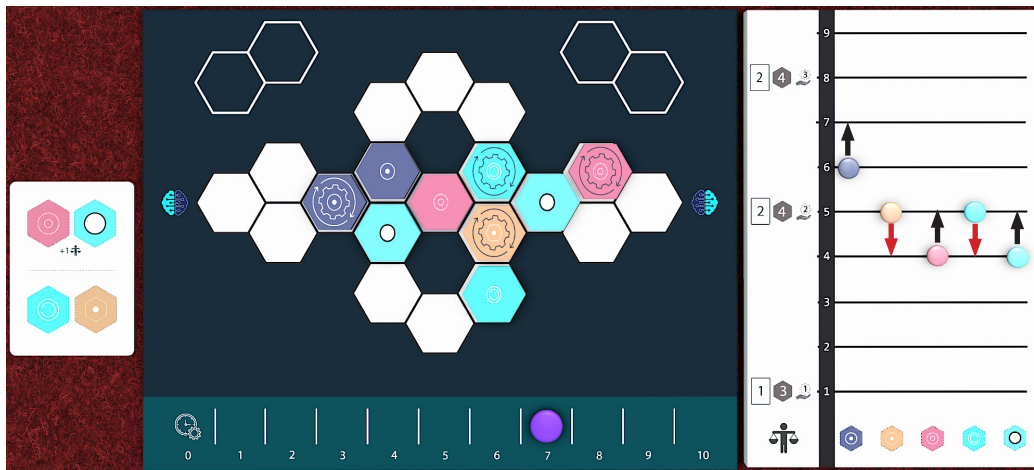



For each type of tile that moves up on the confidence board, on the player board flip **one** tile of that type **face-down**. Flipped (face-down) tiles are not counted in the next Backpropagation (unless flipped back using an Operation card).











Important Thematic Explanation: In real-life AI training, backpropagation involves the AI automatically adjusting its own weight values in response to data, and checking its own predictions for accuracy. This is called “training”, which over time makes the AI's predictions and understanding more and more accurate.

However, over-reliance on a particular set of training data can lead to an AI that is smart when shown things it has seen before, but very poor at making decisions about new information. This is called “overfitting”, and in the game is represented by a situation in which your board has weight values that begin to spread apart, with some very high and some very low. You don't want this!

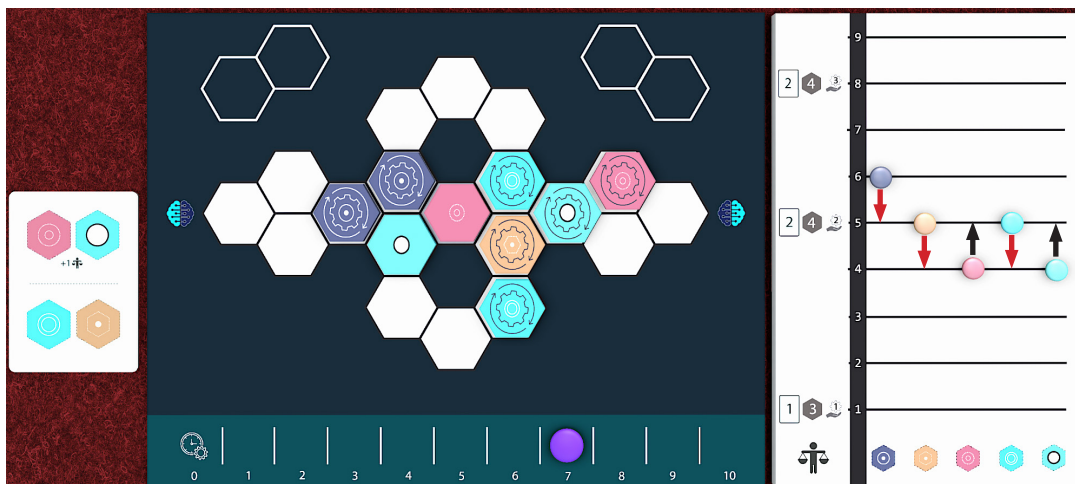


Example 1: The type of tile that has the highest number of unflipped of its type is  with two unflipped tiles. Thus it will move up.

The next three: , , and  are all tied with one unflipped tile each. Thus those 3 must be tie-broken using the order on the bottom of the confidence board. Thus  and  will move up.

Thus , , and  form the top 3 and their weight tokens will move up, and the others will move down.

For each type moved up on the confidence board, **one** of those types of tiles must be flipped face-down (player's choice).



Example 2: There are only two types of tile that are unflipped:  and  so they will both move up and the rest will move down.

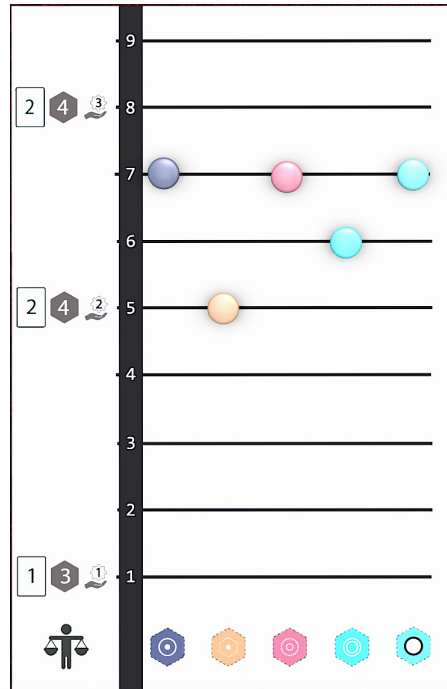
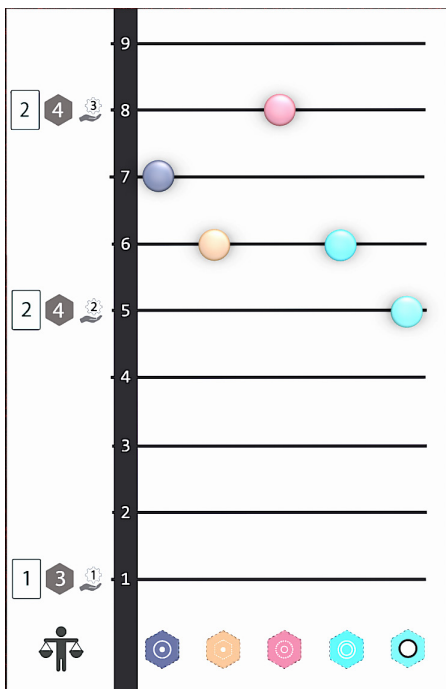
For each type moved up on the confidence board, **one** of those types of tiles must be flipped face-down (player's choice).

NOTE: If there is a 5-way tie for the number of unflipped tiles per type (as in, either there are no unflipped tiles, or all 5 types have the same number of unflipped tiles) then all 5 weight markers move down one space.

Game End

After all players finish backpropagation, a new round is started in the new player order. Once the last tile is drawn from the draw pile, the game immediately ends and the following occurs:

1. All players take one final action (if they can afford it) in player turn order, starting with the player after the player who drew the last tile. If they cannot afford their last action, or they choose not to take one, they are skipped. If tiles are drawn using the Download Data action, they are not replaced on the action board.
2. All players perform **one final backpropagation**. No income is taken.
3. A winner is determined. All players compare how high up on their confidence board their **lowest** weight token is. If there is a tie, the tied players then compare the position of their second-lowest weight token, then third-lowest, etc. until a winner is determined. If all 5 weight tokens are tied, then the player who has the most GPU cycles wins.



Example: The above two players compare their confidence boards. Each has their lowest weight token at confidence level 5. Thus they are tied so far. They both have their next lowest token on level 6. Still a tie.

However, the first player has *another* token on level 6, and the second player has their next token on level 7, so the second player wins! Notice that the second player won, even though the first player had the token in the highest overall position.



Solo Mode

A solo mode is in development. Meanwhile, the game is fairly easy to play two-handed, which is a great way to learn to play the game. Thanks for playing!

- Tony