TTIC 31230, Fundamentals of Deep Learning

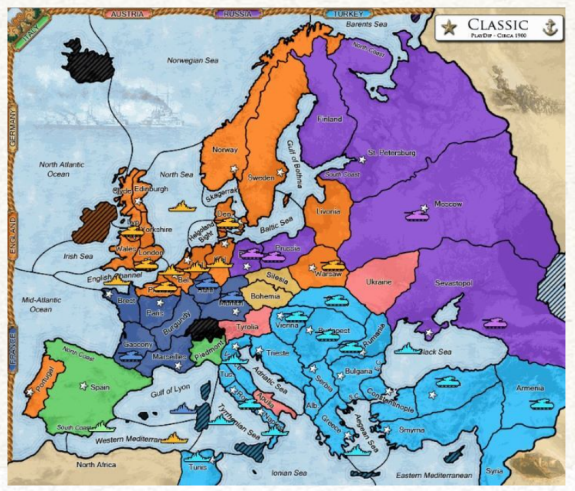
David McAllester, Autumn 2024

Diplomacy

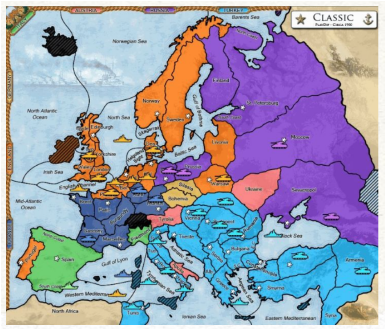
Cicero Plays Diplomacy

Human-level play in the game of Diplomacy by combining language models with strategic reasoning, (Meta, November 2022)

Players engage in secret negotiaions to conspire to take over the world.



Cicero Plays Diplomacy



The play consists of a sequence of "turns". In each turn each player converses with the others using a private messaging system. After some time the messaging ends and each player submits orders to their generals. The orders are revealed simultaneously. Promises can be broken. The orders determine a state transition on the game board (no dice).

Cicero Plays Diplomacy

Cicero is an AI agent that plays Diplomacy with humans. It is rated in the top 10% of human players.

Overall Comments

The natural language aspect of the game turns out be the easy part.

In the game of Diplomacy the set of possible orders that can be given for a given board configuration is highly constrained.

This greatly facilitates interpreting and generating language.

The hard part is "solving" the formal game (selecting negotiation proposals and final orders).

GPT-4 Cannot Play Formal Games

It is worth noting that GPT-4 cannot play chess or Diplomacy — it seems GPT-4 cannot effectively plan actions in games defined by formal rules.

Game Theory: Nash Equilibria

The game theory of multi-player games, even without negotiation, is complicated.

(Without negotiation) a Nash equilibrium is a policy π_i for each player i such that each player is playing best-response under the policies of the other players.

$$\pi_i = \underset{\pi_i}{\text{argmax}}\, V_i(\pi_i, \pi_{-i})$$

Here π_{-i} is the set of policies for all players other than i.

Planning in a Multi-Player Game

One can plan in a multi-player game by assuming that the players will all find the same Nash equilibrium and that one can find this equilibrium by searching for it.

One then plays one's own best response in the Nash equilibrium.

A form of equilibrium search is done in Cicero.

Negotiation: Correlated Nash Equilibria

At the end of negotiation it is natural to search for policies $\pi_1, \ldots, \pi_n$ that form a Nash equilibrium conditioned on the text of the negotiation.

In the presence of negotiation (or other partially observable "signal") this is called a correlated equilibrium because the negotiation introduces correlations between the actions of different players.

Imitation Learning

Cicero uses both an imitation-trained model and a self-play-trained model.

For imitation learning Cicero uses a dataset of over 12 million games from WebDiplomacy including messages exchanged between players. (WebDiplomacy deidentified Player accounts).

Imitation Learning

We define an "action" to be a set of orders given to generals. The state transition is determined by an end-of-turn action for each player.

Cicero continuously predicts all player's final actions during the negotiation phase (including its own).

$$\text{im}^* = \underset{\text{im}}{\text{argmin}} \; E_{(x_i,a_j)\sim\text{Train}} \left[-\ln P_{\text{im}}(a_j|x_i) \right]$$

"im" stands for "imitation".

Imitation Learning

$$\text{im}^* = \underset{\text{im}}{\text{argmin}}\ E_{(x_i,a_j)\sim\text{Train}}\left[-\ln P_{\text{im}}(a_j|x_i)\right]$$

x_i is taken from a time during negotiation and consists of all past board positions and the previous messages in this negotiation visible to player i.

a_j is the action actually taken at the end of that round by player j.

If $j = i$ this is an imitation action policy for j.

If $j \neq i$ this is i's imitation belief about j intended action.

Learning from Self Play

Cicero trains a value function that assigns a value $V_i(b)$ giving an estimate of the value that player i receives given board position b.

As games are played the value function can be trained with standard methods — Bellman error and replay buffers.

Learning from Self Play

During self-play Cicero is playing all of the players.

Before sending each negotiation message Cicero samples 30 actions for each player i from the imitation action predictions $P(a_i|x_c)$.

Cicero then searches for "equalibrium policies" $\pi_1, \ldots \pi_n$ where each π_i is selecting one of the 30 selected actions for that agent.

Learning from Self Play

Cicero then optimizes policies $\pi_1, \ldots, \pi_n$ according to a piKL best-response objective where π_{-i} is the collection of policies other than π_i.

$$\pi_i^* = \underset{\pi_i}{\text{argmax}}\ V_i(\pi_i, \pi_{-i}) - \lambda KL(\pi_i, P_{\text{im}}(a_i|x_c))$$

At the end of the negotiation each player draws an action from their policy.

Playing with Humans

During negotiation Cicero uses self play to estimate the policies (intentions) of the Human players.

The final actions are (of course) made by Cicero and the human players.

Imitation Message Generation

Cicero has a model $P_{\text{mess}}(y|s, r, a_s, a_r, x_s)$ where where y is the English text of the message, s is the message sender, r is the message recipient, a_s is the action taken by the message sender and a_r is the action taken by the message receiver at the end of the turn (in the future).

x_s consists of all past board positions and the previous messages visible to the sender in the current negotiation.

Imitation Message Generation

$$\text{mess}^* = \underset{\text{mess}}{\text{argmin}} \; E_{\{(x,s,r,y,a_s,a_r)\sim\text{Train}\}} \left[-\ln P_{\text{mess}}(y|s,r,a_s,a_r,x_s) \right]$$

Overall Comments

The natural language aspect of the game turns out be the easy part.

In the game of Diplomacy the set of possible orders that can be given for a given board configuration is highly constrained.

This greatly facilitates interpreting and generating language.

The hard part is "solving" the formal game (selecting negotiation proposals and final orders).

END