

Sztuczna inteligencja. Ostatni wykład on-line

Paweł Rychlikowski

Instytut Informatyki UWr

22 czerwca 2022

Na wykładzie 15:

- Jest o sieciach Bayesowskich, czyli acyklicznych grafach zależnych zmiennych losowych, opisujących rzeczywistość
- Przykładową siecią może być **Model Językowy**
- (na przykład taki, w którym mówimy, że kolejny znak zależy od 3 poprzednich – i tylko od nich)

Algorytm:

- Weź 3 znaki ($c_1 c_2 c_3$) i wylosuj znak c_4
- Weź 3 znaki ($c_2 c_3 c_4$) i wylosuj znak c_5
- i tak dalej

Sydneš bývala v se, nejsem němí před se pracuji, každý musím třídil to, co mít
schodili jsem Green už všemusedla cigarelefon ve nám vzal jsme šokonců bezce poř
ádkakupile je v té vůbecky se zná učím neko pro noc mléčné poda nemoc byla sobil
a a to zábal názor, že se hlavír zpívala dneměl byl pokaždé tro

- Żeby je dokładnie omówić potrzeba innego przedmiotu
- (na przykład Neural Network and Natural Language Processing)

Podstawowa zasada

Model określa prawdopodobieństwo $P(w_t | w_1 \dots w_{t-1})$ – i tym samym daje możliwość generacji tekstów

- **PapuGaPT-2** jest przykładowym, nowoczesnym modelem językowym, do wczytania ze strony Hugging Face
- Generacja tekstów jest bardzo łatwa!

```
from transformers import pipeline, set_seed
set_seed(42)
generator = pipeline('text-generation',
                     model='flax-community/papuGaPT2')
print (generator('I wtedy nagle')['generated_text'])
```

Lubię truskawki z

- bita śmietaną i syropami, jak i klasyczne truskawki z czekoladą, ale też z truskawkami, a
- cukrem. Są pyszne. Na ich bazie są takie lekkie jak budyń i w dodatku bezglutenowe.
- innymi owocami” są mi bliskie do tej pory, ale nie lubię tego określenia. To określenie
- konfiturą z borówki. Na blogu znajdziecie kilka przepisów z nimi w roli głównej. Dziś proponuję
- owocami, ale nie lubię ich zbyt wiele :) Pozdrawiam! Asiu - jeśli chodzi o owoce

Nie cierpię truskawek

- , a to ciasto jest obłędne. W środku jest w ogóle takie trochę, jakby to
- :) I to nie byle jakich - jagoda kamsimonella to moja ulubiona truskawkowa:) aha, bo
- ? O co chodzi? Na co? Czy moje dziecko może jeść truskawki? Czy musi jeść
- a one mi owocują już od kilku miesięcy. Teraz jak patrzę jaki napuchły, zmały tak
- z powodu tak złej pogody i kiepskiego zaopatrzenia rynku, więc nie miałam innego wyjścia, musiałam
- z truskawkami i z cukrem:) Tak naprawdę truskawki nie muszą być kwaśne! Na samą myśl

Pomnóż te dwie liczby, a następnie

- wpisz je w punktach na podstawie wzoru, który masz przed sobą. Powodzenia! Zestawienia te prezentują
- wpisz w odpowiednie miejsce liczby znajdujące się w polu powyżej, a w drugiej wpisz wynik
- wygraj pieniądze na naszym konkursie! Za pomocą myszki wysił się, by wybrać zwycięzcę i pomóc
- pomnóż je. Czy potrafisz, aby na koniec każdy z podanych wyników był dodatni? A jeśli
- podziel przez dwie cyfry. Niech suma obiegów będzie liczbą ujemną, zaś suma wszystkich par dodatnich

Wykłady ze sztucznej inteligencji były

- w tym roku naprawdę niezłe. W dodatku w dwóch przypadkach to był jedyny wykład na
- jedną z moich największych życiowych porażek. Uświadomiłeś sobie, jak ważne w dzisiejszych czasach jest zapewnienie
- też ważnym kontekstem dla wydarzeń związanych z wprowadzeniem technologii i Internetu Rzeczy na rynek w
- okazją do lepszego poznania technologii i procesów chemicznych. A potem na miejscu zwiedzaliśmy sale wystawowe
- bardzo ważne, dla rozwoju i rozwoju człowieka. Nie sposób nie wymienić tu badań nad tymi
- jednym z ciekawszych i oryginalnych osiągnięć szkoły, która po raz pierwszy weszła na wyższy szczebel

- Daniel Kahneman w swoich książkach (na przykład 'Pułapki myślenia. O myśleniu szybkim i wolnym' rozróżnia dwa 'podsystemy mózgu'
- **System 1**: heurystyczny, szybki, przydatny, ale skłonny do pewnego rodzaju błędów
- **System 2**: wolny, leniwy, analityczny, ...

Uwaga

Wydaje się, że to co powyżej widzieliśmy pasuje do Systemu 1 (w przeciwieństwie do większości wcześniejszych wykładów)

Ale czy na pewno?

Wszystko jest tekstem

Jako tekst możemy traktować:

- Zapis nutowy jakiego utworu
- Dowód twierdzenia w jakimś formalnym języku
- Program komputerowy
- Sekwencja stanów gry zręcznościowej albo stanów z Sokobana
- ...

GATO – A Generalist Agent



Figure 1 | **A generalist agent.** Gato can sense and act with different embodiments across a wide range of environments using a single neural network with the same set of weights. Gato was trained on 604 distinct tasks with varying modalities, observations and action specifications.

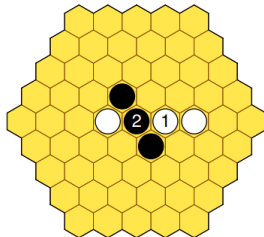
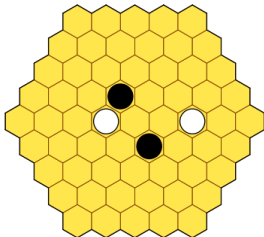
- Na wykładzie 15 mówimy o aukcji Vickreya
- oraz o pierwszej grze planszowej utworzonej przez komputer

Gra Yavalath

Kod gry

```
(game Yavalath
  (players White Black)
  (board (tiling hex) (shape hex) (size 5))
  (end (All win (in-a-row 4)) (All lose (in-a-row 3)))
)
```

Przykładowa sytuacja:



- Ostatnio pismo Communication of ACM wydrukowało artykuł pt. Reimagining Chess with AlphaZero
- Pytania:
 - Czy szachy można poprawić?
 - Czy inne wersje są ciekawsze?
 - Jak warianty gry zmieniają wartość bierek?

Podstawowa metoda

- 1 AlphaZero umie nauczyć się grać w dowolną grę, grając sam ze sobą.
- 2 Możemy zatem nauczyć go grać w **szachy z pięcioma skoczkami i słoniem** (lub w cokolwiek innego)
- 3 a następnie obserwować rozgrywki i wyciągać wnioski

AI is driving the next evolution of chess, giving players a glimpse into the game's future.

**BY NENAD TOMAŠEV, ULRICH PAQUET,
DEMIS HASSABIS, AND VLADIMIR KRAMNIK**

Reimagining Chess with AlphaZero

MODERN CHESS IS the culmination of centuries of experience, as well as an evolutionary sequence of rule adjustments from its inception in the 6th century to the modern rules we know today.¹⁷ While classical chess still captivates the minds of millions of players worldwide, the game is anything but static. Many variants have been proposed and played over the

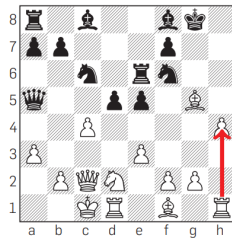
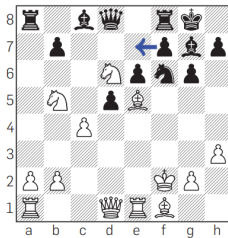
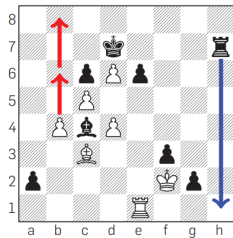
game and is unlikely to ever fall out of fashion, alternative variants provide an avenue for more creative play. In Fischer random chess, the brainchild of former world champion Bobby Fischer, the initial position is randomized to counter the dominance of opening preparation in a game.⁷ One could consider not only entirely new ideas, but also reassess some of the newer additions to the game. For example, the “castling” move was only introduced in its current form in the 17th century. What would chess have been like had castling not been incorporated into the rules? Without recourse to repeating history, we reimagine chess and address such questions in silico with AlphaZero.²⁵

AlphaZero is a system that can learn superhuman chess strategies from scratch without any human supervision.^{19,22} It represents a milestone in artificial intelligence (AI), a field that has ventured down the corridors of chess more than once in search of challenges and inspiration. Throughout the history of computer chess, the focus was on creating systems that could spar with top human players over the board.³ Computer chess has progressed steadily since the 1950s, with better-tuned evaluation functions and enhanced search algorithms deployed on increasingly more computational



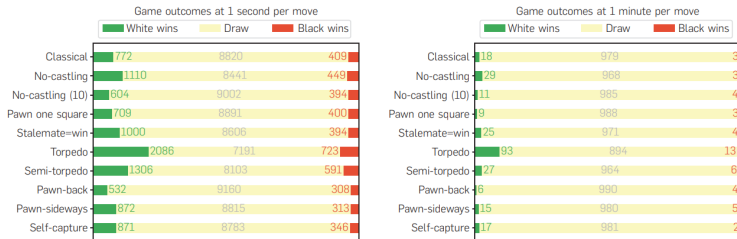
O szachach na nowo: przykładowe nowe ruchy

Figure 1. Examples by AlphaZero of three of the nine chess variants analyzed in this article. In Torpedo chess (left), White generates rapid counterplay with a torpedo move (b4-b6). ♖h1 is followed by yet another torpedo move, b6-b8=♖. In Pawn-sideways chess (center), Black plays a tactical sideways pawn move (f7-e7) after sacrificing a knight on f2 in the previous move, opening the f-file toward the White king. In Self-capture chess (right), White's self-capture move (♙xh4) generates threats against the Black king.



O szachach na nowo: rezultaty rozgrywek

Figure 2. AlphaZero self-play game outcomes: for 10,000 games played at 1 sec per move (left) and for 1,000 games played at 1 min per move (right).













O szachach na nowo: wartości bierek

Metoda

- 1 Przeprowadzać wiele rozgrywek szybkimi agentami, zapisywać przebieg
- 2 Uczyć liniowy klasyfikator przewidywać, czy sytuacja jest zwycięska na podstawie sumarycznej wartości bierek na planszy

Table 3. Estimated piece values from AlphaZero self-play games for each variant.

Variant					
Classical	1	3.05	3.33	5.63	9.5
No-castling	1	2.97	3.13	5.02	9.49
No-castling (10)	1	3.14	3.40	5.37	9.85
Pawn one square	1	2.95	3.14	5.36	9.62
Stalemate = win	1	2.95	3.13	4.76	8.96

Variant					
Self-capture	1	3.10	3.22	5.34	9.42
Pawn-back	1	2.65	2.85	4.67	9.39
Semi-torpedo	1	2.72	2.95	4.69	8.3
Torpedo	1	2.25	2.46	3.58	7.12
Pawn-sideways	1	1.8	1.98	2.99	5.92

- Mówiliśmy o logice pierwszego rzędu, ale nie podaliśmy żadnego mechanizmu wnioskowania w tej logice
- Czas to nadrobić!

Przykład (bardzo stary)

Przesłanki

- Ludzie są śmiertelni, $(\forall x)\text{człowiek}(x) \rightarrow \text{śmiertelny}(x)$
- Sokrates jest człowiekiem, $\text{człowiek}(\text{Sokrates})$

Wniosek

- Sokrates jest śmiertelny, $\text{śmiertelny}(\text{Sokrates})$

- Do wnioskowania z takimi zdaniami potrzebne jest nam **podstawienie**
- (aby otrzymać zdanie bez zmiennych: $\text{człowiek}(\text{Sokrates}) \rightarrow \text{śmiertelny}(\text{Sokrates})$)

Podstawienie

Funkcja która zmiennym przypisuje **termy**, na przykład stałe, takie jak Sokrates, albo bardziej skomplikowane wyrażenia jak na przykład ojciec(Sokrates).

- Podstawienie: $[x=\text{Sokrates}]$ **unifikuje** $\text{człowiek}(x)$ z $\text{człowiek}(\text{Sokrates})$
- Podstawienie: $[x=\text{Hera}, y=\text{Zeus}]$ **unifikuje** $\text{są_parą}(x, \text{Zeus})$ z $\text{są_parą}(\text{Hera}, y)$

```
function UNIFY( $x, y, \theta = \text{empty}$ ) returns a substitution to make  $x$  and  $y$  identical, or failure  
  if  $\theta = \text{failure}$  then return failure  
  else if  $x = y$  then return  $\theta$   
  else if VARIABLE?( $x$ ) then return UNIFY-VAR( $x, y, \theta$ )  
  else if VARIABLE?( $y$ ) then return UNIFY-VAR( $y, x, \theta$ )  
  else if COMPOUND?( $x$ ) and COMPOUND?( $y$ ) then  
    return UNIFY(ARGS( $x$ ), ARGS( $y$ ), UNIFY(OP( $x$ ), OP( $y$ ),  $\theta$ ))  
  else if LIST?( $x$ ) and LIST?( $y$ ) then  
    return UNIFY(REST( $x$ ), REST( $y$ ), UNIFY(FIRST( $x$ ), FIRST( $y$ ),  $\theta$ ))  
  else return failure  
  
function UNIFY-VAR( $var, x, \theta$ ) returns a substitution  
  if  $\{var/val\} \in \theta$  for some  $val$  then return UNIFY( $val, x, \theta$ )  
  else if  $\{x/val\} \in \theta$  for some  $val$  then return UNIFY( $var, val, \theta$ )  
  else if OCCUR-CHECK?( $var, x$ ) then return failure  
  else return add  $\{var/x\}$  to  $\theta$ 
```

Figure 9.1 The unification algorithm. The arguments x and y can be any expression: a constant or variable, or a compound expression such as a complex sentence or term, or a list of expressions. The argument θ is a substitution, initially the empty substitution, but with $\{var/val\}$ pairs added to it as we recurse through the inputs, comparing the expressions element by element. In a compound expression such as $F(A, B)$, OP(x) field picks out the function symbol F and ARGS(x) field picks out the argument list (A, B) .

Wnioskowanie w logice I-go rzędu

```
function FOL-FC-ASK( $KB, \alpha$ ) returns a substitution or false
  inputs:  $KB$ , the knowledge base, a set of first-order definite clauses
            $\alpha$ , the query, an atomic sentence

  while true do
     $new \leftarrow \{ \}$  // The set of new sentences inferred on each iteration
    for each rule in  $KB$  do
       $(p_1 \wedge \dots \wedge p_n \Rightarrow q) \leftarrow \text{STANDARDIZE-VARIABLES}(\text{rule})$ 
      for each  $\theta$  such that  $\text{SUBST}(\theta, p_1 \wedge \dots \wedge p_n) = \text{SUBST}(\theta, p'_1 \wedge \dots \wedge p'_n)$ 
        for some  $p'_1, \dots, p'_n$  in  $KB$ 
         $q' \leftarrow \text{SUBST}(\theta, q)$ 
        if  $q'$  does not unify with some sentence already in  $KB$  or  $new$  then
          add  $q'$  to  $new$ 
           $\phi \leftarrow \text{UNIFY}(q', \alpha)$ 
          if  $\phi$  is not failure then return  $\phi$ 
    if  $new = \{ \}$  then return false
  add  $new$  to  $KB$ 
```

Figure 9.3 A conceptually straightforward, but inefficient, forward-chaining algorithm. On each iteration, it adds to KB all the atomic sentences that can be inferred in one step from the implication sentences and the atomic sentences already in KB . The function `STANDARDIZE-VARIABLES` replaces all variables in its arguments with new ones that have not been used before.

- Tak samo, jak my uczymy się dowodzić twierdzeń, może to robić program komputerowy
- Również samemu generując dane
- Zobacz praca: **Learning to Prove from Synthetic Theorems**