# Ruch drogowy a sztuczna inteligencja

*Paweł GORA*, Warszawa*

Jest to tekst związany z odczytem wygłoszonym na LVI Szkole Matematyki Poglądowej, *Matematyzacja*, Wola Ducka, sierpień 2017.

*Redakcja*

Cztery miliardy złotych – prawie tyle tracą rocznie kierowcy w siedmiu największych polskich miastach z powodu korków [1]. W innych krajach sytuacja wygląda podobnie, korki na drogach są zmorą mieszkańców niemal wszystkich dużych miast na świecie, powodując opóźnienia, straty czasu i paliwa, stres kierowców, hałas i większe zanieczyszczenie powietrza. Mają więc negatywny wpływ nie tylko na portfele uczestników ruchu, ale również na zdrowie mieszkańców miast – smog, który obecnie w dużym stopniu jest powodowany przez samochody, stanowi realne zagrożenie dla zdrowia i życia ludzi.
W przypadku ruchu drogowego zagrożenie życia i zdrowia nie wynika jedynie ze spalin (co w przyszłości, być może, da się częściowo wyeliminować dzięki pojazdom elektrycznym i hybrydowym, przy jednoczesnym stosowaniu bardziej przyjaznych środowisku źródeł energii niż paliwa), ale również z zagrożeń wypadkami. Rocznie na świecie w wypadkach ginie ok. $1,3$ miliona osób, a ponad 20 milionów zostaje rannych (2). W pewnym sensie przyzwyczailiśmy się już do wypadków drogowych, zaakceptowaliśmy to, że występują i niewiele możemy z tym zrobić. Okazuje się jednak, że dzięki nowym technologiom, w szczególności sztucznej inteligencji, ruch drogowy może stać się w przyszłości znacznie bezpieczniejszy i efektywniejszy.

Przypuszczalnie największy przełom będzie możliwy dzięki pojawieniu się na dużą skalę pojazdów autonomicznych, a więc sterowanych przez program komputerowy, a nie przez człowieka. Mogą one zapewnić znacznie większy poziom bezpieczeństwa, gdyż będą jeździć zgodnie z przepisami, będą miały więcej *sensorów* niż człowiek, te sensory będą korzystać z zaawansowanych metod wizji komputerowej do rozpoznawania otoczenia i mogą być skuteczniejsze niż wzrok człowieka. Czas reakcji takich pojazdów będzie krótszy, nie będą się męczyć i dekoncentrować. Dzięki temu liczba ofiar wypadków może spaść wielokrotnie, a ostatecznym celem jest tzw. „wizja 0", w której ofiar wypadków nie ma wcale.

Pojazdy autonomiczne umożliwią transport osobom niepełnosprawnym, starszym i dzieciom, pracę lub odpoczynek podczas podróży, a także szybsze dostarczanie przesyłek. Na świecie pojawiają się już pilotażowe programy usług autonomicznych taksówek zamawianych przez aplikację w telefonie, które same po nas przyjadą i zawiozą nas do celu bezpiecznie i optymalną trasą. Jest to realizacja idei *mobilności na żądanie*, która może sprawić, że będziemy potrzebowali mniej pojazdów i mniej miejsc parkingowych (a zaoszczędzoną przestrzeń miejską można będzie przeznaczyć na tereny zielone lub inne atrakcje dla mieszkańców).

Pojazdy autonomiczne będą mogły również komunikować się z infrastrukturą drogową i między sobą, aby synchronizować pewne manewry, np. wyprzedzanie i bezkolizyjny przejazd przez skrzyżowanie, przybliżając nas jeszcze bardziej do „wizji 0", oraz poprawiając efektywność ruchu – pojazdy będą mogły bowiem podróżować w zwartych grupach, podobnie jak np. ławice ryb, zapewniając płynny przejazd.

Aby to wszystko było możliwe, potrzebne są zaawansowane badania, m. in. po to, aby wyznaczyć właściwe parametry pojazdów autonomicznych, aby zaprojektować odpowiednią infrastrukturę drogową lub optymalnie zarządzać ruchem. Takie badania przeprowadza się już dziś, zarówno w kontekście pojedynczych pojazdów autonomicznych, jak i w kontekście ruchu drogowego z udziałem wielu pojazdów. Podobne badania w przypadku pojazdów tradycyjnych przeprowadza się już od kilkudziesięciu lat i jest już mnóstwo prac naukowych i raportów technicznych dotyczących badań ruchu drogowego. Ogólnie badania te można podzielić na 4 grupy: modelowanie, analiza, predykcja i optymalizacja. W każdej z tych grup kluczowe znaczenie zaczyna odgrywać fascynująca dziedzina nauki, jaką jest sztuczna inteligencja, która w ostatnich latach przynosi coraz większy przełom, zarówno naukowy, jak i przemysłowy,

*Wydział Matematyki, Informatyki i Mechaniki UW, p.gora@mimuw.edu.pl

w wielu dziedzinach nauki i techniki, sprawiając, że programy komputerowe w coraz większym stopniu otrzymują cechy charakterystyczne dla inteligencji człowieka (lub też znacznie możliwości intelektualne człowieka przekraczające).

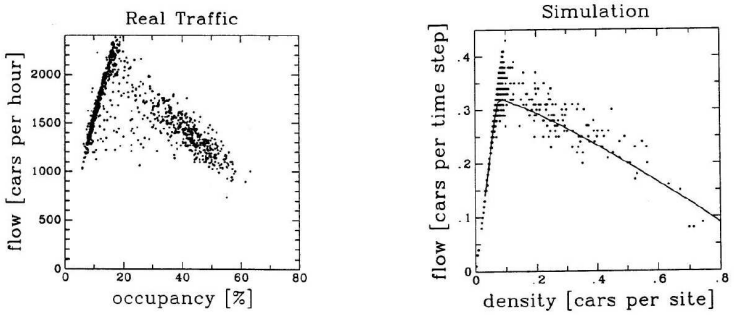
W przypadku modelowania (a więc: budowania matematycznego opisu) zjawiska ruchu drogowego sztuczna inteligencja przydaje się przede wszystkim do kalibracji modeli ruchu, czyli znajdowania takich wartości parametrów rozważanego modelu, aby dawał on jak najlepszą zgodność z obserwowaną rzeczywistością. Do takich zastosowań przydają się m.in. algorytmy klastrowania (np. $k$ najbliższych sąsiadów), algorytmy ewolucyjne, sieci neuronowe, tabu search. Wszystkie te metody uważane są właśnie za przykłady algorytmów sztucznej inteligencji, np. algorytmy klastrowania są w stanie znajdować w dużym zbiorze danych (np. warunków ruchu drogowego) elementy podobne, algorytmy ewolucyjne potrafią efektywnie przeszukiwać olbrzymie przestrzenie możliwych wartości (np. parametrów modeli ruchu), znajdując szybko punkty z przestrzeni, które są dostatecznie dobre według zadanych kryteriów (np. minimalizują wartości pewnej funkcji, np. funkcji odległości danych pochodzących z modelu rzeczywistego od danych dotyczących rzeczywistego ruchu).

Gdy mamy już zbudowany matematyczny model danego zjawiska (w tym przypadku ruchu drogowego), możemy ten model zastosować do analizy tego zjawiska, a potencjalnie również predykcji jego dalszej ewolucji, czy nawet do jego optymalizacji. Metody te (korzystające ze sztucznej inteligencji) zaprezentuję na przykładzie opracowanego przeze mnie modelu ruchu pojazdów w skali dużego miasta, który zaimplementowałem w rozwijanym przeze mnie programie *Traffic Simulation Framework*. Jest on inspirowany dobrze znanym i często badanym modelem Nagela–Schreckenberga (modelem Na-Sch) z 1992 roku [3].

Model Na-Sch bazuje na probabilistycznych automatach komórkowych, które same w sobie nie stanowią jeszcze sztucznej inteligencji, choć automaty komórkowe zostały odkryte przez Johna von Neumanna i Stanisława Ulama w Los Alamos przy okazji badań nad sztucznym życiem i samoreplikującymi się systemami.

W oryginalnym modelu Na-Sch automat komórkowy jest jednowymiarową cykliczną taśmą, podzieloną na komórki odpowiadające rzeczywistej długości $7,5$ metra. Każda z tych komórek może być pusta (jej stan to *null*) lub zajęta (jest na niej pojazd, a stan komórki to aktualna prędkość pojazdu znajdującego się w tej komórce), a ich stan ewoluuje w kolejnych krokach (czas jest dyskretny, prędkości są całkowite) zgodnie z określonymi regułami, odpowiadającymi kolejno przyspieszaniu, zapewnianiu bezpieczeństwa, losowym zaburzeniom (wynikającym np. ze zwiększonej ostrożności) i przemieszczaniu się.

1. $V_i$=min$\{V_i + 1, V_{MAX}\}$ ($V_{MAX}$ – maksymalna dopuszczalna prędkość),
2. $V_i$=min$\{V_i, d_i\}$ ($d_i$ – odległość (liczba komórek) do najbliższego pojazdu przed pojazdem $i$),
3. $V_i$=max$\{V_i - 1, 0\}$ z prawdopodobieństwem $p$,
4. Pojazd $i$ przemieszcza się o $i$ komórek do przodu.



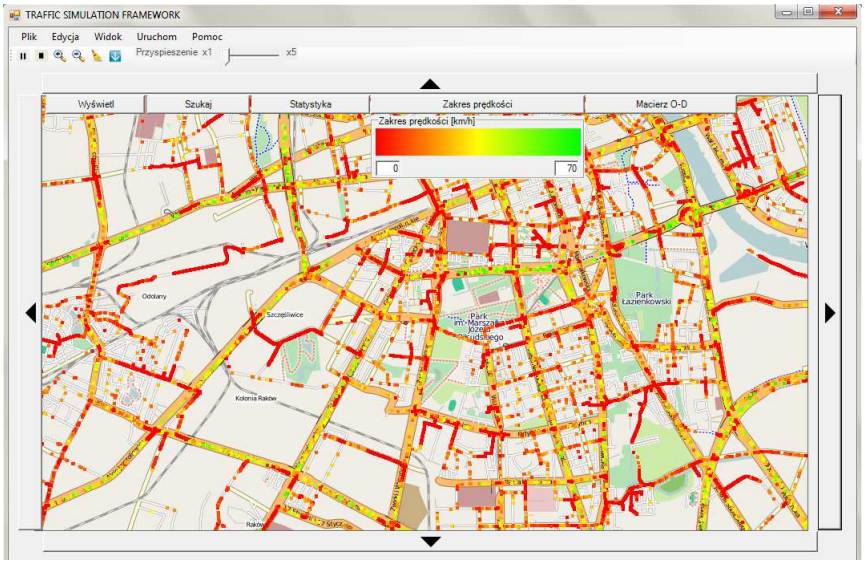

Rys. 1. Porównanie diagramu fundamentalnego dla modelu rzeczywistego ruchu i dla modelu Nagela–Schreckenberga.

Model Nagela–Schreckenberga okazał się bardzo popularny, gdyż pomimo swojej prostoty jest w stanie odzwierciedlać pewne złożone cechy ruchu (np. spontaniczne formowanie się korków na autostradzie) i daje pewną zgodność z rzeczywistością w przypadku diagramu fundamentalnego przedstawiającego zależność między natężeniem ruchu a gęstością pojazdów (rys. 1). Powstało wiele uogólnień tego modelu, m.in. na przypadek dwuwymiarowy i dwa pasy ruchu.

Również ja w swoich badaniach inspirowałem się modelem Na-Sch, ale wprowadziłem istotne zmiany, aby móc symulować ruch na realistycznej sieci drogowej dużego miasta.

W przypadku mojego modelu sieć drogowa jest reprezentowana jako graf skierowany, w którym ruch odbywa się po krawędziach (podzielonych na taśmy – pasy ruchu, podzielone na komórki) zgodnie z regułami zbliżonymi do reguł z modelu Na-Sch – są jednak pewne modyfikacje. W niektórych wierzchołkach grafu znajdują się sygnalizacje świetlne, niektóre wierzchołki reprezentują też zakręty lub skrzyżowania, w związku z tym prędkość pojazdu musi być odpowiednio redukowana przed takimi punktami, możliwe jest też wyprzedzanie. Pojazdy są rozróżnialne – mogą startować z różnych punktów, mieć różne trasy i różny profil określający domyślną maksymalną prędkość pojazdu w zależności od typu drogi. Poza samymi regułami ruchu kluczowe jest też generowanie punktów początkowych i końcowych (służyć do tego mogą macierze podróży, które są budowane w trakcie okresowych pomiarów ruchu) oraz tras przejazdu – do czego przydają się już właśnie algorytmy sztucznej inteligencji: np. heurystyczny algorytm $A^*$ oraz algorytmy kalibracji modeli (minimalizujące różnice między natężeniami ruchu i czasami przejazdu wynikające z modelu oraz z danych rzeczywistych).

Przedstawiony ogólnie model zaimplementowałem w programie komputerowym *Traffic Simulation Framework* (TSF) w języku programowania C#, korzystając z realistycznej sieci drogowej Warszawy pochodzącej z serwisu OpenStreetMap (OSM). Model oraz jego implementacja w programie TSF zostały opisane (bardziej szczegółowo) w moich wcześniejszych pracach [4]. Doczekały się też już pierwszych zastosowań, m.in. w 2010 roku wspólnie z firmą TunedIT współorganizowałem konkurs dotyczący predykcji ruchu drogowego przy prestiżowej konferencji ICDM 2010 [5]. Za pomocą programu TSF wygenerowałem duże ilości danych dotyczących ruchu drogowego w Warszawie, które posłużyły jako zbiory treningowe i testowe dla algorytmów uczenia maszynowego prognozujących ruch (m.in. natężenia ruchu, czasy przejazdu, występowanie korków) w różnych sytuacjach. Uczestnicy konkursu opracowali wiele ciekawych algorytmów korzystających z zaawansowanych technik analizy danych i sztucznej inteligencji, a najlepsze z nich zostały zaprezentowane na konferencji ICDM 2010 w Sydney i opisane w publikacjach.




Rys. 2. Widok graficznego interfejsu użytkownika programu *Traffic Simulation Framework*.

Kolejnym zastosowaniem programu TSF i sztucznej inteligencji jest wykrywanie i analiza stanów ruchu drogowego. Człowiek, obserwując ruch drogowy, jest w stanie dość dobrze ocenić, jaki jest stan tego ruchu (np. czy jest korek, czy ruch swobodny). Zbierając tego typu informacje od wielu obserwatorów ruchu, rejestrując jednocześnie dane sensoryczne (np. położenia i prędkości pojazdów), możemy *nauczyć* programy komputerowe rozpoznawania stanów ruchu drogowego na podstawie danych sensorycznych, aby robiły to podobnie jak ludzie. Może mieć to zastosowanie m. in. w systemach zarządzania ruchem i systemach informowania podróżnych [6].

Gdy potrafimy wykrywać stany ruchu drogowego, możemy pokusić się o krótkoterminową (np. za $10-20$ minut) lub długoterminową (np. następnego dnia) predykcję tych stanów lub bardziej elementarnych parametrów ruchu

(np. czasy przejazdu, natężenia ruchu czy właśnie stany ruchu). Jest już dużo prac naukowych, w których badano algorytmy predykcji ruchu, od stosunkowo elementarnych, jak ARIMA, czy kNN, po bardziej zaawansowane i nowoczesne, jak XGBoost, rekurencyjne sieci neuronowe (np. *long short-term memory* – LSTM [7]), konwolucyjne sieci neuronowe lub rozwiązania hybrydowe stanowiące połączenie tych metod. To właśnie z tematem predykcji ruchu związany był wspomniany konkurs, który współorganizowałem wspólnie z TunedIT, pod patronatem IEEE, ICDM i TomTom [5].

Metody te sprawdzają się dość dobrze w przypadku predykcji ruchu w sytuacjach typowych, tzn. gdy warunki ruchu są typowe, powtarzalne. Gdy obserwujemy ruch drogowy przez dłuższy czas, możemy zauważyć, że warunki ruchu faktycznie dość często są powtarzalne. W poniedziałki rano korki tworzą się w tych samych miejscach, podobnie w szczycie popołudniowym, codziennie są też podobne godziny, w których ruch się uspokaja. Jest to spowodowane powtarzalnością zachowań ludzi, którzy ruch generują (np. w podobnych godzinach rozpoczynają i kończą pracę lub zajęcia w szkole). Analizując dane dotyczące ruchu, można więc zauważyć pewne powtarzalne wzorce. Można jednak również zauważyć pewne fluktuacje – losowe odchylenia od tych wzorców spowodowane np. pogodą, bliskością świąt lub wakacji, imprezami masowymi, nowymi osobami przyjeżdżającymi do miast itp. O ile predykcja ruchu w sytuacjach typowych jest stosunkowo prosta, o tyle prognozy dla sytuacji nietypowych są znacznie trudniejsze. Ciężko bowiem przewidzieć, analizując jedynie historyczne dane, że z ruchu zniknie nagle wiele tysięcy pojazdów, gdy spłonie Most Łazienkowski w Warszawie, a taka sytuacja faktycznie miała miejsce [8].

W takich sytuacjach metodom uczenia maszynowego z pomocą mogą przyjść modele ruchu drogowego i zachowania podróżnych, które w przypadku odpowiedniej kalibracji są w stanie przewidywać, jak ruch może wyglądać. Problemem może być jednak pozyskanie odpowiednio dobrych danych do kalibracji, aby takie analizy przeprowadzać – do tego mogą bardzo przydać się m.in. dane z urządzeń mobilnych, z którymi podróżują ludzie, co pozwala mieć znakomite informacje o zachowaniach podróżnych w różnych sytuacjach. Dane takie, m.in. tzw. *floating car data*, są już coraz bardziej dostępne, z czego korzystam w badaniach dotyczących predykcji ruchu realizowanych w ramach międzynarodowego projektu *MELODIC* [9]. Wprowadzając takie dane do symulacji ruchu drogowego, można nie tylko skalibrować model ruchu, ale również w czasie rzeczywistym uruchamiać symulacje i przeprowadzać za ich pomocą prognozy ruchu (podobne podejście stosowane jest np. na autostradach w Niemczech [10]).

Ciekawym kierunkiem badań związanych z predykcją ruchu jest tzw. predykcja *what-if*: chcemy przewidywać, jak będzie wyglądał ruch w przypadku, gdy coś się zmieni, np. jakie będą czasy czekania pojazdów na czerwonym świetle, gdy zmienimy ustawienie sygnalizacji świetlnej. Naturalne jest przypuszczenie, że ponownie możemy skorzystać z symulacji ruchu drogowego, wystarczy tylko wprowadzić do symulatora inne ustawienia świateł, nie zmieniając innych parametrów (w szczególności pozostawiając te same punkty początkowe, końcowe i trasy pojazdów), co oczywiście jest możliwe.

Widać, że tego typu predykcja może być krokiem w kierunku optymalizacji ruchu drogowego. Gdy potrafimy ewaluować duże ilości różnych ustawień sygnalizacji świetlnych, możemy wybrać najlepsze z nich i zastosować w rzeczywistym ruchu. Problem polega na tym, że możliwych ustawień jest bardzo dużo. Jeżeli przyjmiemy 800 skrzyżowań ze światłami, a w przybliżeniu tyle jest ich w Warszawie, oraz łączny cykl fazy światła zielonego i czerwonego 120 sekund, daje to $120^{800}$ różnych ustawień przesunięć w fazie (zakładając stałe długości fazy światła zielonego i czerwonego, np. odpowiednio 58 sekund i 62 sekundy, a potencjalnie długości te też mogą się zmieniać). Jest to znacznie więcej niż np. szacowana liczba atomów w obserwowalnym Wszechświecie lub więcej niż liczba możliwych rozgrywek w Go [11]. Znajdowanie optymalnego rozwiązania w tak dużym zbiorze to jak szukanie igły w stogu siana! Co gorsza, ewaluacja

Biorąc pod uwagę, że stóg siana ma mniej atomów niż obserwowalny Wszechświat, a igła ma ich więcej niż 1, problem wydaje się być jeszcze trudniejszy, przynajmniej biorąc pod uwagę rozmiar rozpatrywanych zbiorów.

pojedynczego ustawienia świateł również zajmuje sporo czasu, przynajmniej jeśli chcemy to robić za pomocą symulacji komputerowej, która na standardowym komputerze potrzebuje wielu sekund, aby obliczyć jakość danego ustawienia choćby w ciągu kilku minut ruchu. Moglibyśmy mieć nadzieję, że problem da się rozwiązać jakimś sprytnym algorytmem, bez przeszukiwania dużej przestrzeni rozwiązań, udowodniono jednak, że już nawet w bardzo prostych modelach ruchu problem jest NP-trudny.

Sytuacja wydaje się więc beznadziejnie trudna, jednak, być może, nie musimy wyszukiwać w zbiorze możliwych ustawień tego jedynego najlepszego? W końcu symulacyjne modele ruchu są też tylko pewnym przybliżeniem rzeczywistości, więc i tak nie mamy gwarancji, że najlepsze ustawienie świateł według symulatora ruchu będzie również najlepsze w rzeczywistości (w rzeczywistym ruchu może być nawet trudno zdefiniować, czym dokładnie jest optymalność ustawienia świateł). Może w takim razie zadowoli nas rozwiązanie suboptymalne, trochę gorsze od tego najlepszego, trudnego do znalezienia, ale niewiele, np. o 1%, a do tego znacznie łatwiejsze do znalezienia, gdyż takich suboptymalnych rozwiązań jest po prostu znacznie więcej.

Z pomocą mogą przyjść heurystyki, np. algorytm genetyczny inspirowany procesem doboru naturalnego [12]. W naszym przypadku możemy kodować ustawienie sygnalizacji świetlnej do postaci genotypu, wektora genów, wygenerować pewną losową pulę początkową rozwiązań, których jakość może być oceniana za pomocą symulacji ruchu i przeprowadzać ewolucję tej populacji zgodnie z określonymi regułami selekcji, krzyżowania i mutacji genotypów. Procedura ta po pewnym czasie powinna znaleźć odpowiednio dobre rozwiązanie – optimum lokalne. W praktyce takich procedur możemy chcieć przeprowadzić dużo, np. startując z różnych pul populacji początkowych, testując różne ustawienia selekcji, krzyżowania i mutacji itp. Oznacza to, że również dużo musielibyśmy przeprowadzić symulacji ruchu drogowego, co ponownie może być bardzo czasochłonne.

Z problemem tym radzę sobie, stosując tzw. metamodele, a więc przybliżenia matematycznych modeli ruchu. Mogą być to jeszcze prostsze modele, powstałe z tych bardziej skomplikowanych poprzez zastosowanie pewnych założeń upraszczających – w praktyce okazuje się jednak, że trudno jest w ten sposób osiągnąć jednocześnie znaczne przyspieszenie obliczeń i zachować odpowiednią dokładność. Z pomocą mogą przyjść algorytmy uczenia maszynowego, np. sieci neuronowe (inspirowane z kolei działaniem ludzkiego mózgu), które są znane jako dobre uniwersalne aproksymatory funkcji ciągłych na zbiorach zwartych. Tzw. *Twierdzenie o Uniwersalnej Aproksymacj* i mówi o tym, że dowolną funkcję ciągłą na zwartym podzbiorze $\mathbb{R}^n$ można przybliżać jednostajnie z dowolnie dobrą dokładnością sieciami neuronowymi z jedną warstwą ukrytą [13]. Co prawda, twierdzenie nie mówi o tym, jak dużo powinno być neuronów w tej warstwie, jaki powinien być zbiór treningowy i czas trenowania, a nasza funkcja mapująca ustawienia sygnalizacji świetlnej na łączne czasy czekania na czerwonym świetle, obliczana za pomocą symulacji, nie jest ciągła (choć można ją uciąglić), ale twierdzenie sugeruje, że sieci neuronowe warto przynajmniej wypróbować do tego zagadnienia i to również z większą liczbą warstw neuronów (spektakularne sukcesy głębokich sieci neuronowych w ostatnim czasie związane są właśnie m.in. z większą liczbą warstw neuronów).

Pomysł okazał się bardzo dobry. Za pomocą programu TSF przeprowadziłem 117033 symulacji ruchu z różnymi ustawieniami świateł (przesunięć w fazie) na obszarze Starej Ochoty obejmującym 15 skrzyżowań ze światłami. W każdej symulacji obliczany był łączny czas czekania pojazdów na czerwonym świetle na tym samym obszarze w ciągu 10 minut symulacji z udziałem 42000 pojazdów (najdłuższy czas czekania wyniósł 62911 sekund, najkrótszy 35831 sekund – a więc odpowiednio średnio mniej niż $1,5$ i mniej niż 1 sekundę na samochód).

Wygenerowany zbiór wyników został następnie podzielony na zbiór treningowy, walidacyjny i testowy w celu trenowania sieci neuronowych, które mogłyby przybliżać wyniki symulacji. Dlatego przygotowany został program TensorTraffic, korzystający z biblioteki TensorFlow, służącej właśnie m.in. do trenowania sieci neuronowych.

W systematycznie przeprowadzonych eksperymentach trenowane były sieci feed-forward z funkcją aktywacji ReLU i różną liczbą warstw (od 1 do 15 warstw), różną liczbą neuronów w warstwach (zazwyczaj dobre wyniki dawało kilkaset neuronów), różnymi wartościami parametru uczenia i dropout (parametru odpowiedzialnego za losowe usuwanie neuronów z sieci w celu przeciwdziałania przeuczeniu), testowane były też różne rozmiary zbioru treningowego oraz *mechanizm zatrzymania* w sytuacji, gdy wartości funkcji straty użytej do trenowania sieci (RMSE) przestawały się poprawiać.

Najlepsze wyniki udało się otrzymać dla sieci, która miała 3 warstwy (a w nich odpowiednio 100, 200 i 100 neuronów), parametr uczenia $0,01$, dropout $0,05$. Średni błąd dla takiej sieci wyniósł $1,2\%$, a błąd maksymalny $6,8\%$. Okazało się również, że rozkład błędu na zbiorze testowym był dość korzystny, dla większości elementów błąd przybliżeń był stosunkowo mały, poniżej 2%. Po wytrenowaniu sieć neuronowa była w stanie dokonywać ewaluację nowych ustawień świateł w czasie o kilka rzędów wielkości krótszym niż w przypadku symulacji (ok. $0,2$ sekundy).

Dzięki temu można było potem przeprowadzić szybciej dużo eksperymentów ze wspomnianymi algorytmami genetycznymi oraz innymi algorytmami optymalizacyjnymi. Okazało się, że np. już przy populacji początkowej składającej się ze 100 ustawień, po 30 iteracjach udaje się znajdować genotyp lepszy o ok. $20-30\%$ od najlepszego genotypu w początkowej populacji (dla losowo wybranych 10 milionów ustawień najlepsze z nich okazało się gorsze od najlepszego znalezionego przez algorytm genetyczny). Wyniki opisanych badań zostały już opublikowane w 2 pracach naukowych [14, 15], są też cały czas systematycznie ulepszane.

Przedstawiona metoda może w przyszłości posłużyć do konstrukcji nowej generacji systemów zarządzania ruchem, może się również przydać do wielu innych zastosowań związanych z optymalizacją złożonych procesów. W podobny sposób jak ruch drogowy można automatami komórkowymi modelować rozwój nowotworu i przybliżać tę ewolucję w zależności od terminów i dawek podawania radioterapii, podobnie można również znajdować optymalne terminy i dawki za pomocą algorytmu genetycznego. Biorąc pod uwagę, że niektóre automaty komórkowe są Turing-zupełne (np. *Gra w życie*), metoda może również potencjalnie przydać się do znajdowania przybliżonych rozwiązań wielu innych problemów obliczeniowych.

**Referencje**


1. Raport Deloitte: `https://www2.deloitte.com/pl/pl/pages/public-sector/articles/korki-w-polskich-miastach.html`.
2. Raport WHO: `http://www.who.int/news-room/fact-sheets/detail/road\-traffic-injuries`.
3. Nagel, K.; Schreckenberg, M. (1992). *A cellular automaton model for freeway traffic* in Journal de Physique I. 2 (12): 22.
4. Gora P., "*Traffic Simulation Framework – a Cellular Automaton based tool for simulating and investigating real city traffic*, in „Recent Advances in Intelligent Information Systems", 2009, pp. 641-653, ISBN: 978-83-60434-59-8.
5. TomTom Traffic Prediction for Intelligent GPS Navigation: `http://tunedit.org/challenge/IEEE-ICDM-2010`.
6. Wasilewski P., Gora P., *Traffic-related Knowledge Acquired by Interaction with Experts*, in „Proceedings of the Workshop Concurrency, Specification & Programming'2014", Chemnitz, Sept. 29 – Oct. 1, 2014, Informatik-Berichte der HUB Nr. 245, pp. 269–280, 2014.
7. Kang D., Lv Y., Chen Y., *Short-term traffic flow prediction with LSTM recurrent neural network*, in „2017 IEEE 20th International Conference on Intelligent Transportation Systems", `https://ieeexplore.ieee.org/document/8317872`, 2017.
8. Raport na temat liczby podróży `https://tvnwarszawa.tvn24.pl/informacje,news,gdzie-sie-podzialy-52-tysiace-brkierowcow-z-mostu-lazienkowskiego,160863.html`.
9. Strona projektu *MELODIC*: `http://melodic.cloud`.
10. Marinósson S.F., Chrobok R., Pottmeier A., Wahle J., Schreckenberg M., *Simulation Framework for the Autobahn Traffic in North Rhine-Westphalia*, ACRI 2002: Cellular Automata pp. 315–324.
11. *Number of Possible Go Games*: `https://senseis.xmp.net/?NumberOfPossibleGoGames`.
12. Michalewicz Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag.
13. Cybenko, G. (1989), *Approximations by superpositions of sigmoidal functions*, Mathematics of Control, Signals, and Systems, 2 (4), pp. 303–314.
14. Gora P., Kurach K., *Approximating Traffic Simulation using Neural Networks and its Application in Traffic Optimization*, in „NIPS 2016 Workshop on Nonconvex Optimization for Machine Learning: Theory and Practice".
15. Gora P., Bardoński M., *Training neural networks to approximate traffic simulation outcomes*, in „2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), IEEE", pp. 889-–894.