GPS Tour

Filip Mišún

2. mája 2019

1 Zadanie

Cieľom projektu je použiť namerané GPS dáta na natrénovanie HMM, ktoré bude schopné z takýchto dát zistiť, v ktorom momente nositeľ GPS lokátora stojí, kráča alebo jazdí autom.

2 Predspracovanie Dát

Na trénovanie modelu používame reálne dáta namerané GPS lokátorom. Tieto dáta obsahujú informáciu o zemepisnej šírke a dĺžke lokátora ako aj o jeho nadmorskej výške. V modeli, ktorý navrhujem, budeme informáciu o zmenách nadmorskej výšky ignorovať a budú nás zaujímať len zmeny v zemepisnej šírke a dĺžke, ktoré si pre účely inferencie a trénovania "rozbalíme" do dvojrozmernej plochy (náš model teda pracuje len bodmi v dvojrozmernom euklidovskom priestore). GPS lokátor meria pozíciu turistu v časových intervaloch, ktoré nemusia byť úplne pravidelné. Aby sme sa zbavili tejto nepravidelnosti, namerané dáta pred ich použitím interpolujeme, čím dostaneme približnú trasu, po ktorej sa turista pohyboval. Z tejto interpolovanej trasy následne vyvzorkujeme nové dáta, tentoraz v pravidelných časových intervaloch (napr. 20 sekúnd). Takéto pravidelné dvojrozmerné dáta následne používame na inferenciu a trénovanie modelu, ktorý teraz popíšeme.

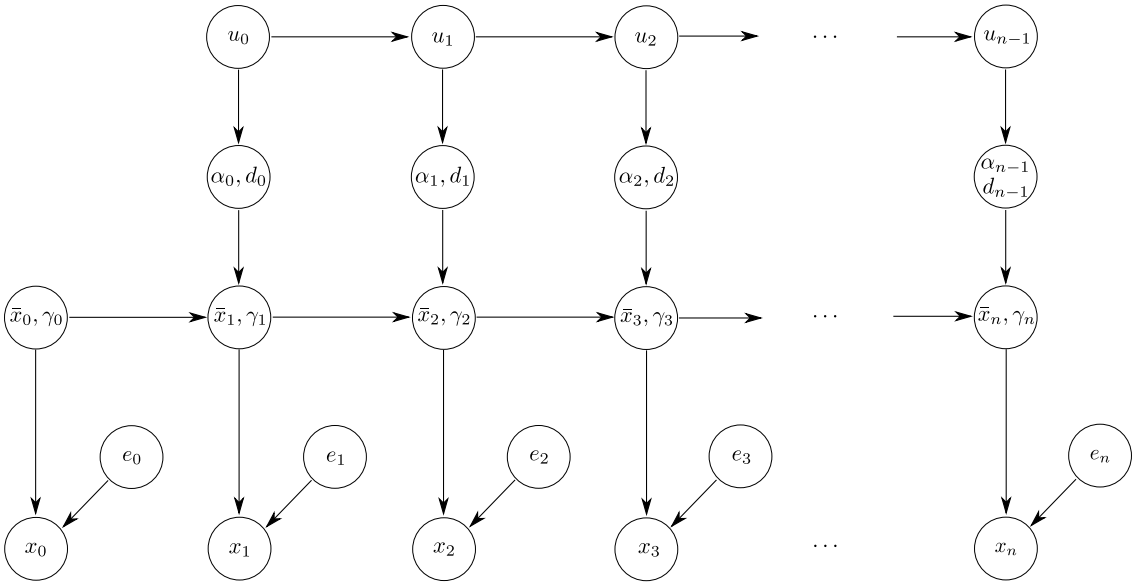
3 Návrh Modelu

Navrhnuté HMM bude pozostávať z troch stavov, ktoré zodpovedajú stavu, v ktorom sa turista (nositeľ GPS lokátora) práve nachádza – stav R ("Rest"), v ktorom turista odpočíva, stav W ("Walk"), v ktorom turista kráča a stav C ("Car"), v ktorom turista jazdí autom. Štruktúru tohto HMM popíšeme najprv neformálne. Každý stav bude generovať vhodne zakódovaný dvojrozmerný vektor, ktorý určuje smer a vzdialenosť, o ktorú sa turista pohne. V stave R budeme očakávať, že pozícia turistu sa bude náhodne pohybovať v nejakom obmedzenom priestore (čo je dané jednak nepresnosťou merania GPS lokátora, ale možno aj pohybom samotného turistu v rámci tohto obmedzeného priestoru). Ak je turista v stave W, teda kráča, očakávame, že generované vektory budú mať tendenciu ísť jedným smerom, t.j. vygenerovaný vektor bude mať väčšinou zhruba rovnaký smer ako predchádzajúci vektor. Taktiež očakávame, že dĺžka vygenerovaných vektorov bude približne zodpovedať priemernej rýchlosti chôdze človeka. Ak turista jazdí autom, teda je v stave C, bude situácia podobná ako v stave W, t.j. vygenerované vektory budú mať tendenciu ísť rovnakým smerom ako predchádzajúci vygenerovaný vektor, avšak v tomto prípade budeme očakávať, že dĺžka vektora bude približne zodpovedať priemernej rýchlosti auta.

Teraz popíšeme model formálne. Navrhnutý model nebude presne zapadať do definície skrytých Markovovských modelov, bude však ich jemnou modifikáciou. Každý stav v našom modeli bude generovať dvojicu reálnych čísel (α, d), kde číslo α reprezentuje uhol, o ktorý sa turista odchýlil oproti svojmu predchádzajúcemu pohybu, kým číslo d bude zodpovedať vzdialenosti, ktorú turista prešiel. Každý stav teda vlastne generuje dvojrozmerný vektor daný polárnymi súradnicami. Čísla α a d budú v skutočnosti spojité náhodné premenné s distribúciou, ktorá bude iná pre každý

stav. V stave W (turista kráča) by mohla mať odchýlka α Gaussovskú distribúciu s nulovou strednou hodnotou a vhodne malou varianciou. Toto zodpovedá našej predstave, že turista bude mať tendenciu ísť približne rovnakým smerom ako pri predošlom meraní smeru chôdze. Za vhodnejší však považujem mix Gaussovskej distribúcie a uniformnej distribúcie na intervale $[-\pi, \pi]$, aby mal turista možnosť raz za čas zmeniť smer svojej chôdze aj výraznejšie. Dĺžka d bude mať Gaussovskú distribúciu s vhodnou strednou hodnotou, ktorá by mohla približne zodpovedať priemernej rýchlosti chôdze človeka, a vhodnou varianciou. V stave C (turista jazdí autom) bude situácia veľmi podobná: náhodná premenná α bude mať distribúciu danú mixom Gaussovskej a uniformnej distribúcie, zatiaľ čo náhodná premenná d bude mať čisto Gaussovskú distribúciu so strendou hodnou, ktorá by mala zodpovedať priemernej rýchlosti auta.

Odlišná situácia je v stave, keď turista odpočíva. Stav R bude akousi idealizovanou realizáciou tejto situácie: keďže sa turista nehýbe, generované premmenné α a d budú vždy nadobúdať hodnotu 0, t.j. premenné α a d majú Diracovu distribúciu so stredom v nulovom bode. Samozrejme, v reálnych dátach budeme v takejto situácii pozorovať šum.



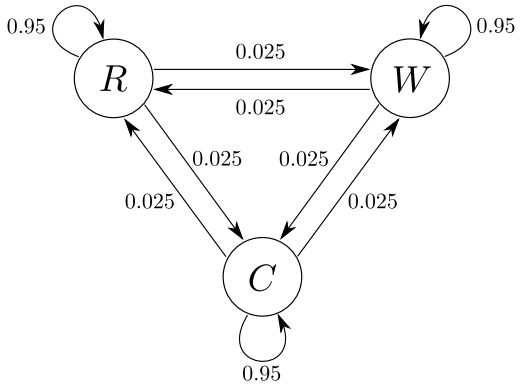
Obr. 1: Diagram Bayesovskej siete nášho modelu.

Náš model bude mať teda podobu, ktorá sa dá výhodne popísať pomocou Bayesovských sietí, tak ako je to znázornené na obr. 1. Vrchný rad premenných u_i v tomto obrázku zodpovedá postupnosti stavov. Každý takýto stav určuje stav nasledujúci, čo je znázornené horizontálnymi šípkami medzi nimi. Vo všeobecnosti platí zásada, že stav u_{i+1} bude chcieť byť rovnaký ako stav u_i, kým prepínanie medzi rôznymi stavmi bude zriedkavé, viď obr. 2. Stavy určujú náhodné premenné α_i a d_i, tak ako to bolo popísané v predošlých odstavcoch. Premenné α_i a d_i následne určujú "skutočnú" pozíciu turistu na povrchu Zeme, ktorú označujeme symbolmi $\bar{x}_i$, a zároveň určujú uhly γ_i udavájúce smer, v ktorom sa turista pohybuje. Uhly γ_i a pozície $\bar{x}_i$ počítame nasledovne

$$\gamma_i = \gamma_{i-1} + \alpha_i,$$
$$\bar{x}_i = \bar{x}_{i-1} + d_i \cdot (\cos\gamma_i\,, \sin\gamma_i).$$

Z týchto vzorcov vidíme, že premenné $\bar{x}_i$ a γ_i závisia od tých istých premenných $\bar{x}_{i-1}$, γ_{i-1} v predchádzajúcom kroku, čo je aj znázornené v obr. 1 horizontálnymi šípkami medzi týmito premennými. Na "skutočné" pozície turistu $\bar{x}_i$ následne pôsobí Gaussovský šum e_i, čím dostaneme "pozorované" pozície turistu x_i, t.j.

$$x_i = \bar{x}_i + e_i.$$



Obr. 2: Prechody medzi stavmi HMM.

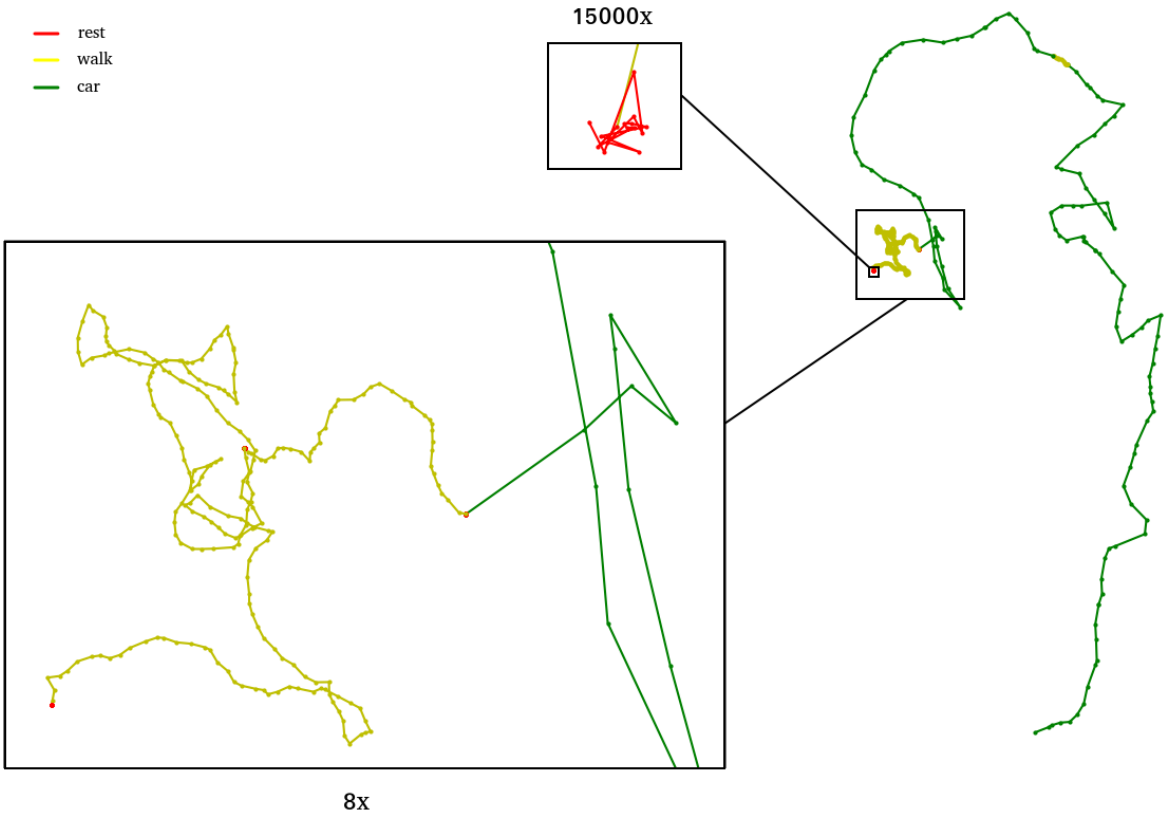
Na obr. 3 je ukážka postupnosti pozícií turistu, ktoré sme dostali náhodným vzorkovaním v našom modeli. Jednotlivé farby v tomto obrázku znázorňujú stav, v ktorom sa turista práve nachádzal, t.j. či odpočíval, kráčal alebo jazdil autom.

4 Inferencia

Z GPS lokátora poznáme postupnosť pozícií turistu $x_0, x_1, \ldots, x_n$, nás však zaujíma predovšetkým postupnosť stavov $u_0, u_1, \ldots, u_{n-1}$, ktorá tieto pozície vygenerovala. Našim cieľom v inferencií bude nájsť postupnosť stavov s najvyššou pravdepodobnosťou alebo aspoň takú postupnosť, v ktorej bude pravdepodobnosťou vygenerovania pozícií $x_0, x_1, \ldots, x_n$ dostatočne vysoká. Takúto postupnosť stavov by sme sa mohli pokúsiť nájsť pomocou algoritmu EM, ktorý by však pre dlhšie postupnosti mohol bežať netriviálne dlho. Namiesto toho odvodíme modifikovaným Viterbiho algoritmom, ktorý nám síce nebude garantovať nájdenie postupnosti stavov s najvyššou pravdepodobnosťou, ale napriek tomu by mal v našej aplikácii dobre zafungovať.

Predpokladajme teda, že poznáme postupnosť pozícií $x_0, x_1, \ldots, x_n$. Ak by sme poznali postupnosť šumov $e_0, e_1, \ldots, e_n$, vedeli by sme určiť postupnosť vygenerovaných uhlov $\alpha_0, \alpha_1, \ldots, \alpha_{n-1}$ a vzdialeností $d_0, d_1, \ldots, d_{n-1}$, z ktorých by sme potom vedeli určiť postupnosť stavov $u_0, u_1, \ldots, u_{n-1}$ použitím Viterbiho algoritmu. Keďže postupnosť šumov $e_0, e_1, \ldots, e_n$ nepoznáme, budeme ju zatiaľ jednoducho ignorovať a postupnosť pozícií $x_0, x_1, \ldots, x_n$ použijeme na odvodenie odhadov uhlov $\hat{\alpha}_0, \hat{\alpha}_1, \ldots, \hat{\alpha}_{n-1}$ a vzdialeností $\hat{d}_0, \hat{d}_1, \ldots, \hat{d}_{n-1}$, ktoré vypočítame tak, ako keby šumy $e_0, e_1, \ldots, e_n$ boli všetky nulové. V prípade, že dĺžky $\hat{d}_{i-1}$ a $\hat{d}_i$ sú výrazne väčšia ako štandardná odchýlka Gaussovského šumu, bude $\hat{d}_i$ aj $\hat{\alpha}_i$ dosť dobrou aproximáciou čísel d_i a α_i, takže ignorovanie chyby e_i možno považovať za opodstatnené. Konkrétne takúto situáciu možno očakávať v prípade, že stav u_i je stavom chôdze alebo stavom jazdy v aute, kedy môžeme pripustiť predpoklad, že Gaussovský šum e_i je jednoducho "pohltený" distribúciou premenných α_i a d_i. Situácia je však odlišná v prípade, že stav u_i je stavom odpočinku, keďže v tomto prípade nemôžeme beztrestne povedať, že šum e_i je pohltený Diracovou distribúciou premenných α_i a d_i vzhľadom na charakter tejto distribúcie. Tento problém by sa dal vyriešiť úpravou modelu, v ktorej by boli distribúcie premenných α_i, d_i upravené tak, že by už v sebe zahŕňali šum e_i. Konkrétne, v stave R by premenná d_i mala Gaussovskú distribúciu s nulovou strednou hodnotou a vhodne malou varianciou, zatiaľ čo premmenná α_i by mala uniformnú distribúciu na intervale $[-\pi, \pi]$. Pri testovaní na validačných dátach sa ukázalo, že takýto model funguje celkom dobre, avšak v niektorých prípadoch môže zlyhávať, pretože pripúšťa presúvanie turistu aj v prípade, že odpočíva (je v stave R).

Pravdepodobnosť postupnosti stavov som sa na nakoniec rozhodol odhadovať nasledujúcim spôsobom. Predpokladajme, že pozorujeme postupnosť pozícií $x_0, x_1, \ldots, x_n$ a chceme spočítať pravdepodobnosť, že táto postupnosť bola vygenerovaná za postupnosti stavov $u_0, u_1, \ldots, u_{n-1}$.



Obr. 3: Názorná ukážka cesty vygenerovanej popísaným HMM. Napravo je znázornená celá vygenerovaná cesta, naľavo sú zväčšené dve časti cesty. Farba vyjadruje stav turistu: červená je odpočinok, žltá je chôdza a zelená je jazda autom. Vyvzorkované dáta majú modelovať situáciu, v ktorej sa meria pozícia turistu na povrchu Zeme každých 20 sekúnd.

Pre túto pravdepodobnosť platí

$$P(u_0, u_1, \ldots, u_{n-1} \mid x_0, x_1, \ldots, x_n) \propto P(x_0, x_1, \ldots, x_n \mid u_0, u_1, \ldots, u_{n-1}) \cdot P(u_0, u_1, \ldots, u_{n-1}).$$

Pravdepodobnosť postupnosti stavov $P(u_0, u_1, \ldots, u_n)$ spočítame štandardným spôsobom, t.j.

$$P(u_0, u_1, \ldots, u_{n-1}) = P(u_0) \cdot P(u_1 \mid u_0) \cdot P(u_2 \mid u_1) \cdot \ldots \cdot P(u_{n-1} \mid u_{n-2}),$$

kde $P(u_{i+1} \mid u_i)$ sú pravdepodobnosti prechodov medzi stavmi, tak ako je to znázornené na obr. 2. Pravdepodobnosť $P(x_0, x_1, \ldots, x_n \mid u_0, u_1, \ldots, u_{n-1})$ odhadneme nasledujúcim spôsobom. Nech I_R a I_{NR} sú množiny indexov takých, že

$$I_R = \{i \in \mathbb{N} \mid u_i = R \ \vee \ u_{i-1} = R\},$$
$$I_{NR} = \{i \in \mathbb{N} \mid u_i \neq R\}.$$

Predpokladajme, že I_R môžeme rozdeliť na k súvislých podpostupností, ktoré budeme označovať $S_1, S_2, \ldots, S_k$. Každá takáto podpostupnosť zodpovedá jednému stanovišťu, v ktorom turista odpočíva. Pozíciu j-teho stanovišťa r_j odhadneme ako

$$r_j = \frac{1}{|S_j|} \sum_{i \in S_j} x_i,$$

t.j. ako ťažisko všetkých pozorovaných pozícií v rámci j-teho stanovišťa. Pravdepodobnosť postupnosti pozícií $x_0, x_1, \ldots, x_n$ za predpokladu postupnosti stavov $u_0, u_2, \ldots, u_{n-1}$ potom odhadneme

ako

$$P(x_0, x_1, \ldots, x_n \mid u_0, u_1, \ldots, u_{n-1}) = \prod_{j=1}^{k} \prod_{i \in S_j} \mathcal{N}(x_i \mid r_j, \sigma^2) \cdot \prod_{i \in I_{NR}} P(\hat{\alpha}_i, \hat{d}_i \mid u_i),$$

kde σ^2 je variancia Gaussovského šumu a $\hat{\alpha}_i$ a $\hat{d}_i$ sú odhady uhlu α_i a vzdialenosti d_i spočítané z postupnosti pozícií $x_0, x_1, \ldots, x_n$. Celkovo teda dostávame

$$P(u_0, \ldots, u_{n-1} \mid x_0, \ldots, x_n) \propto \prod_{j=1}^{k} \prod_{i \in S_j} \mathcal{N}(x_i \mid r_j, \sigma^2) \cdot \prod_{i \in I_{NR}} P(\hat{\alpha}_i, \hat{d}_i \mid u_i) \cdot P(u_1) \cdot \prod_{i=1}^{n-1} P(u_{i+1} \mid u_i).$$

Postupnosť stavov s najväčšou pravdepodobnosťou odhadneme algoritmom, ktorý je na pohľad identický s Viterbiho algoritmom. Tento algoritmus bude vypĺňať tabuľky Path tak, že v $\text{Path}(t, u)$ bude uložená pravdepodobná postupnosť stavov $u_0, u_2, \ldots, u_t$, ktorá generuje postupnosť pozícií $x_0, x_1, \ldots, x_{t+1}$. Ukáže sa, že $\text{Path}(t, u)$ teoreticky nemusí byť najpravdepodobnejšia postupnosť stavov, ale bude sa dať očakávať, že v praxi bude veľmi blízka najpravdepodobnejšej. Tabuľku vypĺňame rekurzívne tak, že postupnosť stavov $\text{Path}(t, u)$ bude rovná postupnosti $\text{Path}(t-1, v)$ pre vhodný stav v, ku ktorej ešte na koniec pridáme stav u. Stav v vyberieme tak, aby pravdepodobnosť postupnosti $\text{Path}(t, u)$ bola najvyššia možná. Formálne, $\text{Path}(t, u)$ nastavíme tak, že

$$\text{Path}(t, u) = \text{Path}(t-1, v) + u,$$

kde v je stav taký, že

$$v = \operatorname*{arg\,max}_{w \in \{R,W,C\}} P\big(\text{Path}(t-1, w) + u \mid x_0, \ldots, x_{t+1}\big),$$

pričom výrazy $\text{Path}(t-1, v) + u$ a $\text{Path}(t-1, w) + u$ treba chápať ako zreťazenia postupnosti stavov $\text{Path}(t-1, v)$, resp. $\text{Path}(t-1, w)$ so stavom u. Ak teda dostaneme postupnosť pozorovaní $x_0, x_1, \ldots, x_n$, tak pravdepodobnú postupnosť stavov odvodíme ako $\text{Path}(n, v)$, kde

$$v = \operatorname*{arg\,max}_{w \in \{R,W,C\}} P\big(\text{Path}(n, w)\big).$$

Vidíme teda, že popísaný algoritmus je vlastne popisom Viterbiho algoritmu. Rozdiel oproti Viterbiho algoritmu je len v tom, ako sa počíta pravdepodobnosť $P(u_0, \ldots, u_t \mid x_0, \ldots, x_{t+1})$ postupnosti stavov $u_0, \ldots, u_t$. Z tohto rozdielu vyplýva aj to, že náš modifikovaný Viterbiho algoritmus nemusí nutne nájsť postupnosť stavov s najvyššou pravdepodobnosťou. Na tento algoritmus sa dá pozerať ako na akýsi greedy algoritmus, v ktorom postupne konštruujeme postupnosť stavov, pričom v každom kroku zoberieme stav, ktorý sa lokálne javí ako najpravdepodobnejší. Dá sa však očakávať, že algoritmus bude v praxi fungovať dobre a navyše, tento algoritmus sa dá implementovať tak, že beží garantovane v lineárnom čase.

5 Trénovanie

Model budeme trénovať algoritmom Viterbiho trénovaním. Náš model najprv vhodne inicializujeme. Následne použijeme popísaný inferenčný algoritmus, aby sme odvodili postupnosti stavov v trénovacích dátach. Zároveň určíme aj postupnosti α_i a d_i a šumy e_i (šum e_i však vieme odvodiť len ak stav u_{i-1} je stavom odpočinku). Takto odvodené parametre modelu použijeme na vyladenie parametrov modelu, konkrétne ladíme:

- Pravdepodobnosti prechodov medzi stavmi R, W, C.
- Varianciu Gaussovského šumu náhodnej premennej e.
- Váhy Gaussovskej a uniformnej zložky premennej α ako aj varianciu Gaussovskej zložky. Ladíme zvlášť α generované stavom W a stavom C.

- Strednú hodnotu a varianciu náhodnej premennej d. Ladíme zvlášť d generované stavom W a stavom C.

Inferenciu a ladenie parametrov modelu opakujeme až do konvergencie.

Dôležitá je pri trénovaní dobrá inicializácia parametrov modelu, ak chceme aby model skonvergoval do prijateľného lokálneho minima. Pri inicializácií modelu využívame svoj "prior knowledge" o problematike. Pravdepodobnosti prechodov medzi stavmi R, W, C inicializujeme tak, že stav sa s vysokou pravdepodobnosťou bude chcieť zachovať a len s malou pravdepodobnosťou sa prepne na iný stav, viď obr. 2. Varianciu Gaussovského šumu premennej e nastavíme na vhodne malú hodnotu (napr. 1 meter). Strednú hodnotu premennej d generovanej stavom W nastavíme na priemernú rýchlosť chôdze človeka, zatiaľ čo strednú hodnotu premennej d generovanej stavom C nastavíme na priemernú rýchlosť auta. Varianciu Gaussovskej zložky premennej α nastavíme na vhodne malú hodnotu, aby mal chodiaci turista, resp. auto tendenciu zachovávať smer. Váhy Gaussovskej a uniformej zložky premennej α nastavíme v prospech Gaussovskej zložky, rovnako z dôvodu, aby bola tendenia zachovávať smer.

6 Vyhodnotenie

Pri vyhodnocovaní modelu nás zaujíma kvalita inferencie postupnosti stavov z danej pozorovanej postupnosti pozícií turistu. Keďže k dátam nemáme "ground truth", fungovanie modelu vyhodnocujem manuálne, prezeraním výstupov natrénovovaného modelu. Výsledky testovania si čitateľ môže prezrieť na `https://github.com/filip-misun/gps_tour/tree/master/kml`. Link odkazuje na repozitár s troma `kml` súbormi, ktoré odporúčam zobraziť v Google Earth. Všetky tri súbory obsahujú dáta z jednej túry, každý z nich však zachytáva pohyb turistu v inom stave, t.j. jeden zachytáva turistu v aute, ďalší zachytáva kráčajúceho turistu a posledný obsahuje dáta o odpočívajúcom turistovi. Dáta z jednotlivých súborov sa v Google Earth zobrazujú odlišnou farbou (jazda autom je zelená, chôdza je žltá a odpočinok je červený).

Výsledky na testovacích dátach vyhodnocujem tak, že model funguje veľmi dobre, no občas sa dopúšťa drobných nepresností. Miestami sa napríklad zdá, že algoritmus necháva vystúpiť turistu z auta o jeden krok skôr ako by mal. Toto môže súvisieť s predspracovaním dát, v ktorom namerané GPS súradnice interpolujeme a vyvzorkujeme v pravideľných intervaloch. Vyvzorkované potom presne nekorešpondujú s pôvodnými dátami, čo môže spôsobovať nepresnosti. Riešením by mohlo byť hustejšie vzorkovanie v interpolovaných dátach, čo by však zase malo za následok pomalšiu inferenciu (v testovacích dátach sa vzorkovanie robilo každých 20 sekúnd).

Hoci v testovacích dátach pozorujem len drobné nedostatky, dá sa očakávať, že v niektorých situáciach by sa náš model mohol dopúšťať aj vážnejších chýb, vyplývajúcich zo značnej jednoduchosti tohto modelu. Ako príklad takéhoto zjednodušovania možno uviesť predpoklad, že rýchlosť jazdy autom možno modelovať unimodálnou normálnou distribúciou. Trpezlivejší študent by mohol navrhnúť iný model, v ktorom by jazda autom bola rozdelená na viacero stavov, v ktorom sú zachytené napr. obmedzenia rýchlosti na rôznych úsekoch cesty. Mohli by sme napríklad zaviesť stavy "jazda v obytnej zóne", "jazda v obci", "jazda mimo obce" a "jazda na diaľnici", v ktorej berieme ohľad na príslušné obmedzenia rýchlosti. Iným riešením by mohla byť úprava distribúcie rýchlosti auta na mix normálnych distribúcií, prípadne na mix normálnych distribúcií a uniformnej distribúcie.

Tak ako to už vo všeobecnosti pri trénovacích algoritmoch býva, ani nášmu modelu by určite neuškodila vhodná regularizácia parametrov modelu, ktorá by bránila vychýleniu parametrov modelu do extrémnych polôh. Autor projektu sa dobrovoľne priznáva, že regularizáciu parametrov neimplementoval len z čírej lenivosti, avšak jedným dychom dodáva, že tento závažný nedostatok sa (na šťastie) na výsledku nepodpísal výrazným spôsobom.

Mojím cieľom však nebolo navrhnúť model, ktorý bude zachytávať všetky nuánsy cestnej dopravy a turistických aktivít všetkého možného druhu. Cieľom bolo skôr navrhnúť funkčný framework, ktorý umožňuje jednoduché rozširovanie a ladenie modelu. Z tohto hľadiska považujem projekt za úspešný (modulo fakt, že nie je implementovaná regularizácia parametrov modelu, čo by sa však dalo bez väčších problémov napraviť).