Modeliranje dinamike fizikalnih sustava naprednim metodama dubokog učenja

Filip Mirković*
Prirodoslovno matematički fakultet,
Fizički odsjek, Sveučilište u Zagrebu,
Bijenička cesta 32, 10000, Zagreb

Mentor: izv. prof. dr. sc. Davor Horatić†

Transformerski su modeli postigli velik uspijeh u različitim zadatcima koji uključuju vremenske nizove te su danas u upotrebi u raznim granama strojog učenja, poput obrade prirodnog jezika i računalnog vida. U ovom radu istražujemo njihov potencijal u modeliranju kaotične dinamike fizikalnih sustava te opisujemo nužne za transformere korake poput učenja vektorske reprezentacije sustava. Transformerom modeliramo Gissinger sustav, koji je uveden kao minimalni model Zemljinog magnetskog polja te posjeduje kaotične izmjene dipolnog momenta.

I. UVOD

Priroda nudi širok asortiman sustava s različitim dinamikama, predviđanje ponašanja tih sustava često je od velikog interesa znanstvenicima i inženjerima. Nažalost mnogi interesantni dinamički sustavi nisu podložni jednostavnom modeliranju pa tako ni predviđanju. Česti uzroci tih poteškoća su visoka dimenzionalnost sustava, nelinearnost te kaotičnost dinamike, također često ni dinamika nije poznata u potpunosti. U praksi najčešće je prisutna kombinacija tih problema.

Duboke neuronske mreže su dobar izbor modela u situacijama kada posjedujemo velik broj podataka o nekom fenomenu. Dapaće, kao univerzalni funkcijski aproksimatori [1], mogu se prilagoditi proizvoljnom skupu podataka. Transformeri spadaju u klasu dubokih modela zasnovanih na mehanizu pozornosti. Smatra se da je tajna njihovog uspjeha pri modeliranju vremenskih nizova leži u korištenju spomenutog mehanizma. Unatoč tome da su transformeri nastali u području obrade prirodnog jezika, njihoa uporaba se proširila na niz drugih područja strojnog učenja. U ovom radu istražujemo njihovu sposobnost u modeliranju dinamike fizikalnih sustava, konkretno radi se o Gissingerovom sustavu Zemljinog magnetizma.

U odjeljku II nudimo kratku definiciju dinamičkih sustava, zatim odjeljak III posvećujemo opisu arhitekture transformera i mehanizma pozornosti. Korz odeljak IV bavimo se učenjem vektorskih reprezentacija fizikalnih sustava te sam Gissingerov sustav će biti detaljnije diskutiran u odjeljku V. Rezultati ovog pokusa nalaze se u odljeljku VI.

II. DINAMIČKI SUSTAVI

Formalno je dinamički sustav uređena trojka prostora stanja $\mathcal{S}$, vremenskog intervala $\mathcal{T}$ te funckcije F. Funckija F definira evoluciju sustava kroz vrijeme, odnosno $F : \mathcal{S} \times \mathcal{T} \longrightarrow \mathcal{S}$. Najčešće je $\mathcal{S} \subseteq \mathbb{R}^n$, npr. u klasičnoj mehanici vektori prostora stanja su vektori generaliziranih koordinata (q, p_q). Ako je vremenski interval kontinuiran $\mathcal{T} \subseteq \mathbb{R}$, F je obična ili parcijalna diferencijalna jednadžba forme

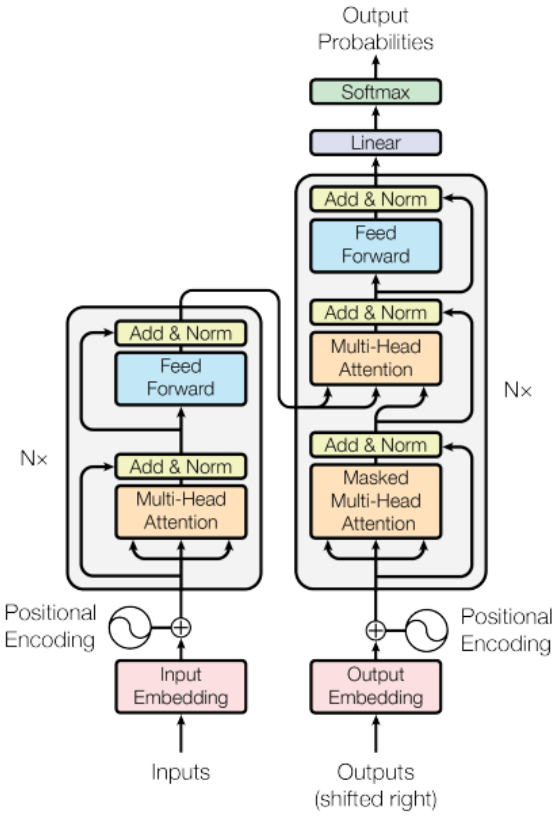
$$\partial_t \boldsymbol{\phi} = F_{\boldsymbol{\eta}}\big(\mathbf{x}, \boldsymbol{\phi}, \nabla_x \boldsymbol{\phi}, \nabla_x^2 \boldsymbol{\phi}, \boldsymbol{\phi} \nabla_x \boldsymbol{\phi}, ...\big) \tag{1}$$

Ovdje je $\boldsymbol{\phi} \in \mathcal{S}$ stanje sustava, $\mathbf{x} \in \mathcal{M}$ točka na mmnogostrukosti na kojoj se sustav opisan (najčešće $\mathbb{R}^m$) te $\eta \in \mathbb{R}^p$ vektor parametara diferencijalne jednadžbe F, a t je parametar duž kojeg pratimo evoluciju sustava, najčešće je vrijeme. Poznavanje dinamike podrazumijeva pronalazak $\boldsymbol{\phi}(t, \mathbf{x})$ za svaki $\mathbf{x}$ iz domene i za svaki t u nekom relevantnom intervalu $\mathcal{T}$. Budući da računala ne mogu obrađivati kontinuirane podatke, prostornu i vremensku domenu diskretiziramo, čime je vremenska evolucija efektivno pretvorena u niz.

III. TRANSFORMERSKI MODEL

Smatramo da bi prije opisa samog modela bilo korisno motivirati njegovu uporabu u modeliranju dinamičkih sustava. Kao što smo već napomenuli, transformeri su originalno nastali u području obrade prirodnog jezika. Prvi zadatci s kojima su se suočili bili su prevođenje ili nadopunjavanje teksta te su se u tim zadatcima bili iznimno uspješni. U takvim se problemima riječi (tokeni) reprezentiraju vektorima. Tu zadaću obvavlja sustav kodera i njegovog inverza dekodera, koji je potpuno odvojen od transformera. Jedino što transformer vidi tijekom treninga i predikcije su vektorske reprezentacije rječi. U tom slučaju, probleme poput prevođenja teksta možemo shvatiti kao učenje preslikavanja jednog niza vektora u drugi s time da među svim vektorima postoji nekakav (vremenski) uređaj. Ovo sugerira da bi transformeri mogli biti uspješni u velikom broju problema koji se mogu formulirati kao traženje preslikavanja jednog niza vektora u drugi [2].

* mirkovic.phy@pmf.hr
† davorh@phy.hr

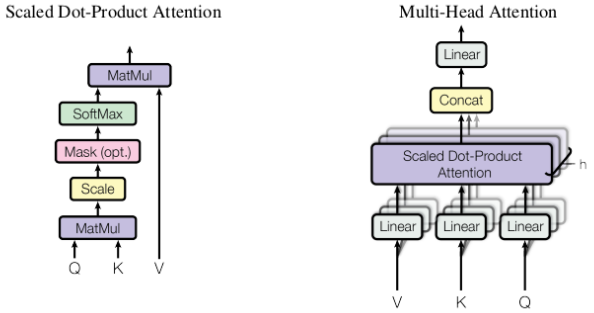


Slika 1. Originalni transformer iz [3], slika je preuzeta iz istog članka. Lijevi blok korespondira transformerskom koderu i on je odbačen u modeliranju dinamičkih sustava. Desni je blok transformerski dekoder koji je upotrebljen za modeliranje dinamike.

Na slici (1) je prikazana arhiektura originalnog transformerskog modela. Ovdje ukratko komentiramo njegove komponente, a zainteresiranog čitatelja referiramo na članke [3] [4]. Originalni model podijeljen je u dva bloka koderski (lijevo) i dekoderski (desno), u modeliranju dinamike koderski blok odbacujemo, no kako se mnogi podblokovi ponavljaju kako u koderskom tako i dekoderskom bloku opis će i dalje ostati cjelovit.

A. Mehanizam pozornosti

Mehanizam pozornosti uveden je nešto prije transformera [5] te se može objasniti u kontekstu preslikavanja niza vektora u drugi niz vektora. U odjeljku [II] objasnili smo kako se računalno modeliranje dinamičkih sustava svodi nastavak niza vektora. Neka je $\Xi = \{\xi_i | i = 1...n\} \subset \mathbb{R}^e$ vremenski uređen niz vektora koji opisuju trenutna stanja promatranog sustava. Ako poznajemo dinamiku, predikciju budućih stanja najčešće možemo obaviti numeričkom integracijom, u tom slučaju jedina informacija potreba iz Ξ jest ξ_n, kojeg koristimo kao



Slika 2. Lijevo: Skalirana funkcija pozornosti. Desno: Višestruka skalirana funkcija pozornosti. Autori A. Vaswani et al. [3] tvrde da funkcija pozornosti bolje funkcionira ako se provodi na h projekcija vektora umjesto na cijelim vektorima, pozornost se dakle računa na h potprostora originalnih vektora. Na kraju se rezultati konkateniraju u vektor originalne dimenzionalnosti.

početni uvjet. Ako dinamika nije poznata te ju modeliramo putem podataka, modelu osim zadnjeg opaženog stanja ξ_n možemo pružiti informaciju o c prethodnih stanja, taj broj c zovemo kontekstna duljina. I dalje smatramo da je za stanje ξ_{n+1} najindikativnije stanje ξ_n, stoga za svaki od c vektora u kontekstom intervalu $C = \{\xi_{n-c}, ..., \xi_n\} \subseteq \mathbb{R}^e$ računamo neku mjeru sličnosti sa stanjem ξ_n. Tu sličnost zovemo pozornost jer nam govori na koja prethodna stanja trebamo najviše uzeti u obzir pri predviđanju budućih stanja. Kao mjeru sličnosti koristimo skalarni produkt normaliziran dimenzijom vektora na ulazu softmax funkcije[3] [2].

U praksi ne računamo skalarni produkt direktno s vektorima iz C, već na vektor ξ_n i vektore iz C djelujemo trima neuronskom mrežama čime dobivamo tri veličine $\boldsymbol{v}, \boldsymbol{q}, \boldsymbol{k}_l \in \mathbb{R}^d$ koje redom zovemo vrijednost ($\boldsymbol{v}$), upit ($\boldsymbol{q}$) i ključevi ($\boldsymbol{k}_l$). Konkretno

$$\boldsymbol{v} = \mathcal{F}_v(\xi_n) \quad \boldsymbol{q} = \mathcal{F}_q(\xi_n) \quad \boldsymbol{k_l} = \mathcal{F}_k(\xi_l) \tag{2}$$

gdje $l = n - c, ..., n$. Tri dodatne neuronske mreže $\mathcal{F}_v$, $\mathcal{F}_q$ te $\mathcal{F}_k$ povećavaju ekspresivnost modela na način da identificiraju bitne značajke u vektorima stanja ξ. Naime, obićan bi skalarni produkt između ξ_n i ξ_l dao svim komponentama vektora jednaku težinu pri određivanju njihove sličnosti, transformiranjem tih vektora u $\boldsymbol{q}$ i $\boldsymbol{k}_l$ omogućavamo modelu da nauči koje su značajke bitne određivanje idućeg stanja.

Sličnosti između ključeva i upita računamo po sljedećem izrazu

$$s_l = \frac{\boldsymbol{q} \cdot \boldsymbol{k_l}}{\sqrt{d}} \tag{3}$$

Skaliranje korijenom dimenzije vektora pomaže numeričkoj stabilnosti pri računanju skalarnog produkta visokodimenzionalnih vektora. Iz mjera sličnosti računamo

koeficijente pozornosti

$$\alpha_l = \text{softmax}(\boldsymbol{s})_l = \frac{e^{s_l}}{\sum_{m=1}^{c} e^{s_m}} \tag{4}$$

Konačno računamo kontekstni vektor kao otežanu sumu svih vektora iz kontekstnog intervala nakon što ih preslikamo u prostor vrijednosti

$$\boldsymbol{c} = \sum_{l=1}^{c} \alpha_l \mathcal{F}_v(\xi_{n-l}) = \sum_{l=1}^{c} \alpha_l \boldsymbol{v}_l \tag{5}$$

Na slici (1) prikazano je kako se obrađuje kontekstsni vektor (izlaz Multi-Head Attention-a). Kontekstni se vektor dodaje originalnom vektoru ξ_n te se provodi normalizacija. Ideja iza uspješnosti ovog pristupa jest da kontekstni vektor $\boldsymbol{c}$ sadrži informaciju o nekim dakekosežnim vremenskim ovisnostima. Izlaz pozornosti se predaje još jednoj potpuno povezanoj neuronskoj mreži.

B. Paralelizacija, maskiranje i kodiranje vremenskog uređaja

U prethodnom odjeljku opisali smo ideju iza mehanizma pozornosti, međutim u praksi postoji par preinaka kojima možemo velik broj operacija paralelizirati, čime se treniranje modela ubrzava. To ubrzanje dolazi pod cijenu uvođenja tzv. maskiranja i kodiranja vremenskog uređaja. Ideja je da u jednadžbi (2) odmah izračunamo i upite i vrijednosti za sve vektore u kontekstnom intervalu te računamo sličnosti između svih parova ključeva i upita čime dobivamo matricu sličnosti i matricu pozornosti

$$S_{lm} = \frac{\boldsymbol{q}_l \cdot \boldsymbol{k}_m}{\sqrt{d}} \tag{6}$$

$$\alpha_{lm} = \text{softmax}(S_{lm}) \tag{7}$$

Na kraju ćemo dobiti c kontekstnih vektora

$$\boldsymbol{c}_l = \sum_{m=1}^{c} \alpha_{lm} \boldsymbol{v}_m \quad l = 1, ..., c \tag{8}$$

Formalno nismo napravili nikakvu promjenu, osim što ovim putem možemo sve operacije izraziti matričnim množenjem koje se mogu efikasno obaviti na grafičkim karticama. Preostaju dva problema. Prvi je da modelu tijekom treniranja ne smijemo dopustiti da obraća pozornost na buduće vektore, konkretno ako predviđamo vektor ξ_{n+1}, zadnji vektor na kojeg smijemo obratiti pozornost je ξ_n. To možemo riješiti ručnom manipulacijom matrice sličnosti (6), tako da sve članove iznad glavne dijagonale postavimo na $-\infty$. Time će po jednadžbi (7) pozornost na tim pozicijama biti 0. Drugi problem jest što model obrađuje sve podatke iz kontekstnog intervala paralelno, čime se gubi informacija o njihovom vremenskom uređaju. Vremenski se uređaj ponovno ostvaruje dodavanjem trigonometrijskih funkcija razlčitim vektorima u kontekstnom intervalu [3] [2]. Simbol $PE_{pos,i}$ označava pozicijsko kodiranje i-te komponente vektora na poziciji *pos* u konekstnom intevalu. Parnim komponentama dodajemo vrijednosti sinusa, a neparnim kosinusa.

$$PE_{pos,2j} = \sin\big(pos/10000^{\frac{2j}{e}}\big)$$
$$PE_{pos,2j+1} = \cos\big(pos/10000^{\frac{2j+1}{e}}\big)$$

C. Funkcija gubitka

Na slici (1) konačni izlaz transformera je softmax. Takva inačica transformera radi nad rečenicama tako da predviđa vjerojatnost iduće riječi u nizu. Sve su riječi unaprijed definirane unutar nekog vokabulara. Budući da je prostor stanja dinamičkih sustava kontinuiran, taj izlaz nije primjeren za ovaj problem. U ovom radu koristili smo transformer koji na izlazu ima linearan sloj [2]. Negativna log-izglednost transformera jest

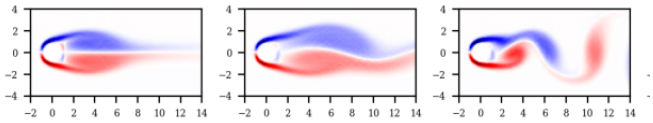
$$\mathcal{L} = -\sum_{i}^{D}\sum_{j}^{c} \log\big(p(\xi_j^i | \xi_{j-c}^i, ..., \xi_{j-1}^i, \theta)\big) \tag{9}$$

Prva se sumacija obavlja po svim kontekstnim intervalima u skupu podataka D. Druga se sumacija provodi po *sljedećem* kontekstnom intervalu odnosno po vektorima koje model mora predvidjeti, on je u gotovo isti kao ulazni kontekstni interval samo što je pomaknut za jedan vremenski korak unaprijed. Budući da želimo modelirati kontinuirane podatke, izglednost modeliramo kao Gausijan između podataka i predviđanja transformera [2]. Time problem svodimo minimiziranje kvadrata 2-norme razlike predikcije i podataka.

$$\mathcal{L} = \sum_{i}^{D}\sum_{j}^{c} ||\xi_j^i - \widehat{\xi}_j^i||_2^2 \tag{10}$$

IV. VEKTORSKA REPREZENTACIJA SUSTAVA

Učenje vektorskih reprezentacija riječi nužna je kako bismo dani problem rješavali dubokim modelima, međutim ta potreba ostaje prisutna pri modeliranju dinamičkih sustava. Ona nije najizraženija kada je sustav prirodno opisan vektorom (n-torkom realnih brojeva) npr. Lorenzov sustav ili dvostruko njihalo, međutim često su sustavi od interesa polja i time posjeduju i prostornu ovisnost. Kako bismo bolje razumjeli potrebu za učenjem vektorskih reprezentacija, zamislimo da promatramo jednu komponentu brzine fluida koji teče preko cilindra kao na slici (3). U računalu će dvodimenzionalno polje strujanja biti reprezentirano matricama $u_{i,j}^t$ gdje $i = 1, ..., N$, a $j = 1, ..., M$, $t =, 1..., T$. Tok Newtonskog fluida je opisan Navier-Stokesovom parcijalnom diferencijalnom jednadžbom zbog čega će najbliži susjedi u



Slika 3. Iznos brzine fluida koji teče preko cilindra u tri uzastopna trenutka kao primjer fizikalnog polja. Preuzeto iz [2].

matrici $u^t_{i,j}$ utjecati na međusobnu dinamiku, npr. $u^t_{i,j}$ i $u^t_{i,j+1}$.

Recimo da, naivno, $u^t_{i,j}$ zapišemo u formi vektora stupca tako da matricu razvučemo po drugoj dimenziji. U tom slučaju točke $u^t_{i,j}$ i $u^t_{i,j+1}$ postaju udaljene za N redova u vektoru, time uništavamo lokalnost koja je bila inherentna matrici $u^t_{i,j}$. Izlaz iz ovog problema jest učenje vektorske reprezentacije danog sustava.

A. Koder i dekoder vektorskih reprezentacija

Krećemo iz osmotrivih (mjerenih) stanja koje opisuju dinamički sustav $\boldsymbol{\phi}_i \in \mathbb{F}$ zatim definiramo kodersku funkciju $\mathcal{F} : \mathbb{F} \to \Xi$ u obliku neuronske mreže. Čest izbor slojeva su konvolucije, budući da njima model uči lokalne značajke podataka [6]. Osim koderske funkcije potreba nam je i dekoderska funkcija $\mathcal{G}$ koja treba što bolje aproksimirati inverz $\mathcal{F}$, odnosno $\mathcal{G} : \Xi \to \mathbb{F}$. Kada je moguće, strukturu dekodera formiramo kao zrcalnu sliku koderske [7]. Ovakav par mreža nije dovoljan za stvaranje smislene reprezentacije vektora, naime ništa ne priječi ove dvije mreže da stanja $\boldsymbol{\phi}_i$ nasumično ne razbacaju po latentnom prostoru Ξ. Želja nam je da originalna dinamika bude prisutna i u latentnom prostoru. To postižemo tako da bliska stanja u vremenu $\boldsymbol{\phi}_i$ i $\boldsymbol{\phi}_{i+1}$ preslikamo u reprezentacije ξ_i i ξ_{i+1} koje su međusobno bliske, s obzirom na euklidsku normu. Za to nam je potreban Koopmanov operator.

B. Koopmanov operator

Ako želimo stvoriti korisnu reprezentaciju stanja sustava kombinacijom kodera i dekodera, latentni prostor trebamo nekako urediti. Teorija Koopmanvih operatora [8] već je neko vrijeme korištena u kombinaciji s dubokim modelima u kontekstu učenja dinamike [9] [10] [11]. Detaljan pregled tog prodrujča bio bi preopsežan za ovaj rad, stoga navodimo intuiciju i osnovne ideje, a za formalne detalje teorije čitatelja referiramo na [8] [2].

Pretpostavljamo da dinamički sustav, bio linearan ili nelinearan, promatramo putem nekog osmotrivog skupa opservabli $\mathbb{F}$ te da dinamika postaje linearna u nekom beskonačno dimenzionalnom prostoru reprezentacija Ξ (U praksi ne možemo baratati s beskonačno dimenzionalnim sustavima, stoga je u primjeru s Gissingerovim sustavom [V] prostor reprezentacija ima 32 dimenzije). U drugim riječima, vremensku evoluciju možemo generirati grupnim translacijama

$$\xi(t) = e^{\mathcal{H}t}\xi(t_0) \tag{11}$$

Kako bismo dobili infinitezimalne translacije sustava gornju jednadžbu razvijamo do prvog člana korekcija u vremenu $t = t_0 + \delta t$.

$$\xi(t + \delta t) = \big(1 + \mathcal{H}\delta t\big)\xi(t_0) = \mathcal{K}\xi(t_0) \tag{12}$$

Iz (12) definiramo Koopmanov operator kao $\mathcal{K}$, kao generator infinitezimalnih vremenskih translacija. U praksi vremenski pomaci su mali, no konačni $\delta t \to \Delta t$, time Koopmanov operator radi diskretno preslikavanje iz ξ_i u ξ_{i+1}. Za n-tu točku u vremenu Koopmanov operator primjenjujemo n puta.

$$\xi_n = \mathcal{K}^n\xi_0 \tag{13}$$

Iz (12) vadimo dva zaključka: prvi da Koopmanov operator ovisi o diskretizaciji vremena, te drugi da je jednak jediničnoj matrici do na korekcije reda $\mathcal{O}(\Delta t)$. Budući da Koopmanov operator učimo iz podataka, navedeni zaključci nas navode na adekvatan oblik Koopmanove matrice: dijagonalni oblik na kojeg dodajemo male korekcije, zatim taj oblik zadržavamo korštenjem L2 regularizacije.

C. Učenje vektorskih reprezentacija

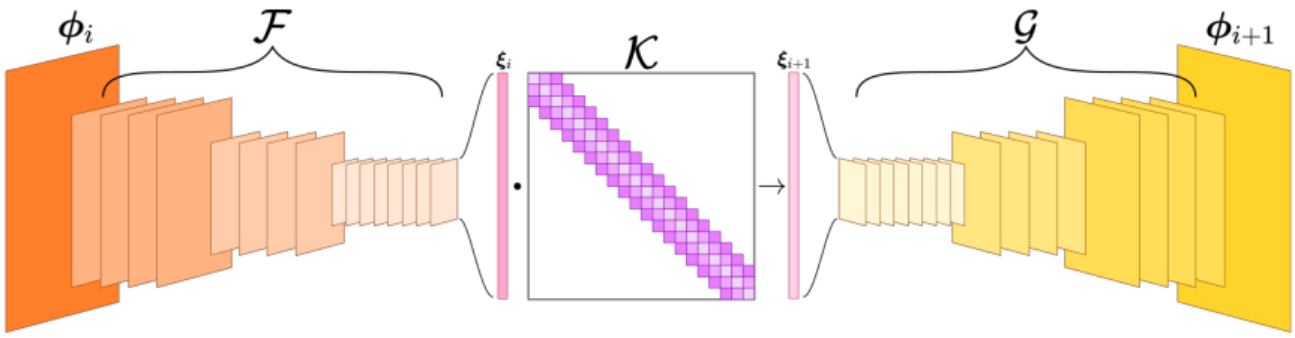
Kao što smo spomenuli potrebne su nam dvije neuronske mreže i jedna prilagodljiva matrica za učenje vektorskih reprezentacija dinamičkih sustava. Zadaće su im generiranje reprezentacija, invertiranje reprezentacija i konačno uređivanje latentnog prostora. Ove tri mreže trenirane su neovisno o transformeru s prikladnom funkcijom gubitka koju rastavljamo na tri djela: rekonstrukcijski gubitak $\mathcal{L}_R$, dinamički gubitak $\mathcal{L}_D$ te regularizacijski gubitak $\mathcal{L}_{reg}$.[2].

$$\mathcal{L}_R = \sum_{i=1}^{D}\sum_{j=0}^{T} \mathrm{MSE}\big(\boldsymbol{\phi}^i_j, \mathcal{G} \circ \mathcal{F}(\boldsymbol{\phi}^i_j)\big) \tag{14}$$

$$\mathcal{L}_D = \sum_{i=1}^{D}\sum_{j=1}^{T} \mathrm{MSE}\big(\boldsymbol{\phi}^i_j, \mathcal{G} \circ \mathcal{K}^j\mathcal{F}(\boldsymbol{\phi}^i_0)\big) \tag{15}$$

$$\mathcal{L}_{reg} = ||\mathcal{K}||_2^2 \tag{16}$$

U jednadžbama iznad koristimo funkciju razlike kvadrata $\mathrm{MSE}(y, \hat{y}) = ||y - \hat{y}||_2^2$, koja se za vektorske podatka svodi na 2-normu razlike vektora primjera i predviđanja. Rekonstrukcijski gubitak osigurava činjenicu da su $\mathcal{F}$ i $\mathcal{G}$ aproksimativno inverzni. Dinamički gubitak zaslužen je



Slika 4. Osmotrivo stanje ϕ_i dinamičkog sustava preslikavamo koderom $\mathcal{F}$ u prostor vektorskih reprezentacija $\xi_i \in \mathbb{R}^e$. Inverz operaciji $\mathcal{F}$ jest dekoder $\mathcal{G}$ koji rekonstruira osmotrivo stanje ϕ_i. Konačno na prostoru vektorskih reprezenacija uvodimo tzv. Koopmanov operator $\mathcal{K}$, koji generira infinitezimalne transformacije u vremenu. Preuzeto iz [2].

za uređivanje latentnog prostora, odnosno želimo da iterativna primjena Koopmanovog operatora ispravno generira sukcesivna stanja sustava. Regularizacijski gubitak prisiljava Koopmanov operator na očekivanu formu infinitezimalnog operatora, koju smo opisali u [IV B]. Znamo da L2 regularizacija matrice parametara ostavlja elemente na glavnoj dijagonali najizraženijima te značajno smanjuje elemente daleko od dijagonale [12]. Konačno potpuni gubitak je

$$\mathcal{L} = \mathcal{L}_R + \mathcal{L}_D + \lambda \mathcal{L}_{reg} \tag{17}$$

Hiperparametar λ određuje snagu te regularizacije.

V. GISSINGEROV SUSTAV

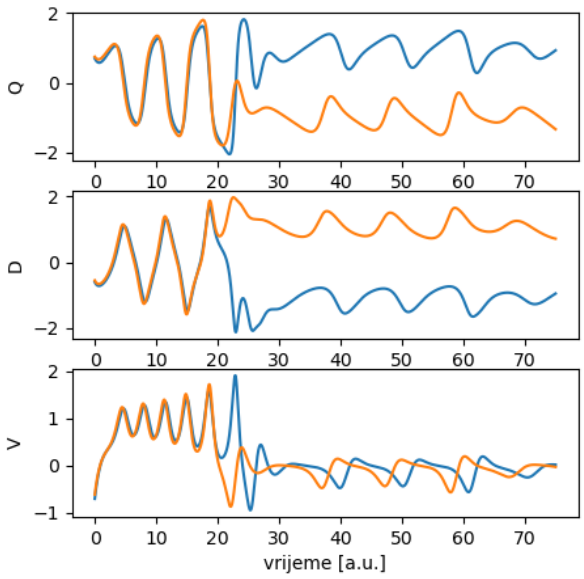
Gissingerov je sustav pronašao Christophe Gissinger u svom članku iz 2012. godine gdje je predložen kao minimalan model za opis obrtaja Zemljinih magnetskih polova[13]. Konkretnije radi se o pojednostavljenim jednadžbama magneto-hidrodinamike

$$\dot{Q} = \mu Q - VD \tag{18}$$

$$\dot{D} = -\nu D + VQ \tag{19}$$

$$\dot{V} = \Gamma - V + QD \tag{20}$$

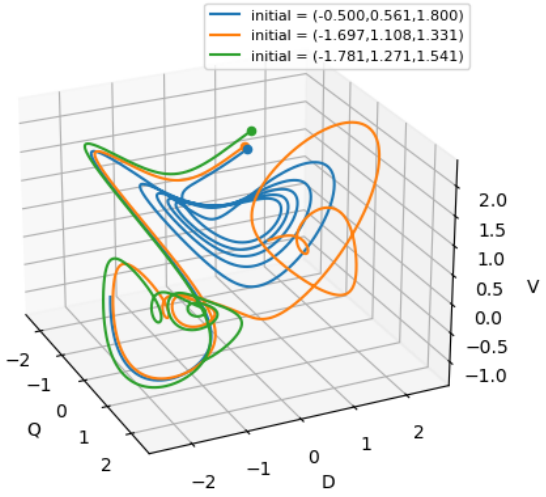
Stanje je sustava opisano trima *modovima*: kvadrupolni Q, dipolni D te brzinski V. Unatoč činjenici da je prejednostavan za precizan opis dinamike magnetskog polja poput Zemljinog, Gissingerov sustav ima zanimljivu i kompleksnu dinamiku koja za određene kombinacije parametara μ, ν i Γ postaje kaotična. Parametri za generiranje podataka odgovarali su kaotičnom režimu sustava i iznose $\mu = 0.119$, $\nu = 0.1$, $\Gamma = 0.9$.



Slika 5. Primjer divergencije trajektorija Gissingerovog sustava. Trajektorije kreću iz bliskih početnih uvijeta te se razdvajaju na vremenskoj skali dva Lyapunovljeva vremena.

A. Lyapunovljev eksponent

Atraktori kaotičnih sustava su karakteristični po tome što je divergencija bliskih krivulja eksponencijalna u vremenu. Lyapunovljev je eksponent korisna veličina pri analizi divergencije trajektorija u faznom prostoru nekog nelinearnog sustava. Intuitivno daje nam mjeru deformacije infinitezimalnog volumena koji okružuje danu točku faznog prostora na atraktoru. Očekujemo de će prije ili kasnije predviđanje modela divergirati od stvarnih trajektorija, to je zbog kaotičnosti samog sustava te činjenice da neuronske mreže same postaju kaotične [14]. Lyapunovljev eksponent, odnosno njegov inverz daje nam ka-



Slika 6. Tri trajektorije Gissingerova sustava za bliske početne uvijete razotkrivaju bogatu dinamiku tog sustava. Naraðasta krivulja je obavavila jednu izmjenu *pola*. Vrijeme integracije 40 [a.u.].

rakterističnu vremensku skalu u kojem sustav postaje nepredvidiv te njime možemo ocijeniti prati li model promatrani sustav dovoljno dugo. Neka je stanje sustava opisano varijablama x_μ $\mu = 1, ..., n$, te njegova neka je njegova dinamika dana s

$$\dot{x_\mu} = f_\mu(\boldsymbol{x}) \tag{21}$$

Dodamo li poznatoj krivulji infinitezimalnu smetnju te dinamičke jednažbe razvijemo u red do linearnog člana, dobivamo jednadžbu kojom možemo pratiti evoluciju te smetnje

$$\dot{x_\mu} + \delta\dot{x}_\mu = f_\mu(\boldsymbol{x} + \delta\boldsymbol{x}) \approx f_\mu(\boldsymbol{x}) + \sum_\nu \partial_\nu f_\mu(\boldsymbol{x})\delta x_\nu \tag{22}$$

$$\delta\dot{x}_\mu = \sum_\nu \partial_\nu f_\mu(\boldsymbol{x})\delta x_\nu \tag{23}$$

Zatim divergenciju trajektorija možemo ocijeniti veličinom [15]:

$$d(t) = \sqrt{\sum_\nu \delta x_\nu(t)^2} \tag{24}$$

Srednji Lyapunovljev eksponent definiran je kao

$$\lambda = \lim_{d(0)\to 0} \lim_{t\to\infty} \frac{1}{t} \log\left(\frac{d(t)}{d(0)}\right) \tag{25}$$

Valja napomenuti da je ovo srednja veličina divergencije trajektorija koja karakterizira atraktor te da različite točke faznog prostora posjeduju različite divergencije trajektorija. Za Gisssingerov sustav jakobijan iznosi:

$$J = [\partial_\nu f_\mu] = \begin{pmatrix} \mu & -V & -D \\ V & -\nu & Q \\ D & Q & -1 \end{pmatrix} \tag{26}$$

Srednji Lyapunovljev eksponent je izračunat prema postupku opisanom u [16]. Integrirano je 150 krivulja u vremenskom intervalu $T = 200$, koji je podijeljen na podintervale duljine $t = 1$. Magnituda inicijalne perturbacije početnog uvjeta iznosi $d(0) = 10^{-2}$. Dobiven je srednji Lyapunovljev koeficijent $\lambda = 0.075$ te je karakteristično vrijeme kaotiziranja $\tau_L = 13.4$.

VI. REZULTATI

Detalji procesa treniranja i generiranja podataka opisani su u dodatcima [A] i [B].

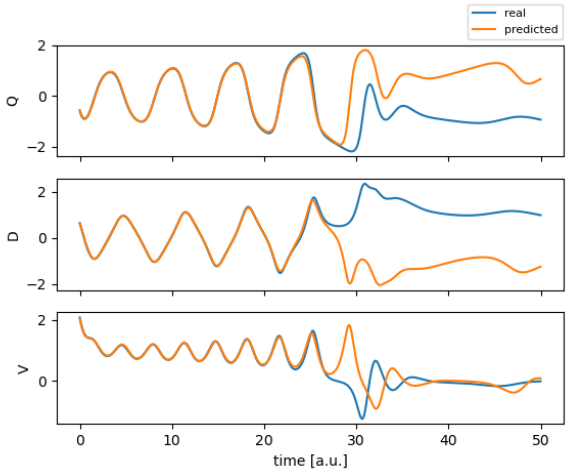
Kod za definiciju i treniranje modela te za vizualizaciju rezultata [2] je prilagođen za Gissingerov sustav. Prilagođen kod je dostupan na `https://github.com/mirxonius/Gissinger_Transformer`.

Unatoč ekspresivnosti modela i dugom treningu, nakon određenog vremena kaos prevlada i predikcije postaju netočne. Ipak općenite karakteristike dinamike su vidljie su u predikcijama modela. Osim toga valja napomenuti da je model treniran na trajektorijama koje su integrirane u $15 \approx 1.12\tau_L$ jedinica vremena, a većina predikcija ostaje precizno na skali dva Lyapunovljeva vremena, ponekad i dulje. Smatramo ovo dobrim rezultatom pri modeliranju sustava s kaotičnom dinamikom. U dodatcima C nudimo još rezultata koji pokazuju kako uspješnost modela uvelike ovisi o izboru početnih uvjeta. Dobili smo predikcije koje ostaju precizne iznimno dugo (slike (11), (9)) te koje brzo divergiraju od stvarnog rješenja (slika (10).

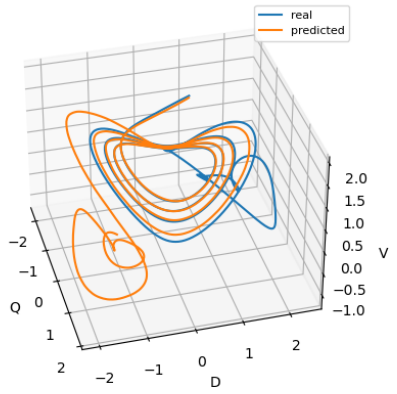
U odsustvu stvarnih podataka nismo ni sigurni koliko je numerička integracija otporna na kaos. Bilo bi zanimljivo usporediti predviđanja numeričke integracije i transforemera sa stvarnim podatcima kaotičnog sustava.

VII. ZAKLJUČAK I DISKUSIJA

U ovom smo radu objasnili postupak modeliranja dinamike fizikalnih sustava transformerskim modelom te primijenili opisan postupak na Gissingerov sustav. Transformeri su se ovdje pokazali kao skupo, ali uspješno rješenje. Uspješno jer su u većini slučajeva daju točna predviđanja na vremenskim skalama koje su usporedive s dvostrukim Lyapunovljevim vremenom. Rješenje je skupo jer zahtijeva puno podataka o problemu te duge i računalno zahtjevne treninge. Također nije jasno koliko greške u kodiranju vektorskih reprezentacija podataka utječu na performanse transformera. U budućim



Slika 7. Usporedba stvarnih trajektorija i predikcije modela. Predviđanja postaju neprecizna na vremenskoj skali 2 Lyapunovljeva vremena. 3D graf istih krivulja je na slici (8)



Slika 8. 3D graf krivulja sa slike (7).

radovima moglo bi se istražiti spoj arhitektura kodera i transformera ili pak dodatno korekcijsko treniranje oba modela zajedno (*eng.* fine tuning). Osim toga moglo bi se isprobati više dubokih modela na Gissingerovom sustavu te usporediti njihovu uspješnost s onom od transformera.

Dodatak A: Treniranje i hiperparametri

Prvo se trenira sustav kodera, dekodera i Koppmanovog operatora stohastičkim gradijentnim spustom uz Adam optimizator. Veličina minigrupe je 512. Inicijalna stopa učenja je $\epsilon = 10^{-3}$, s time da je korišten eksponencijalni prilagođivač (*eng.* scheduler) stope učenja s parametrom $\gamma = 0.955$, koji regulira brzinu pada stope učenja kroz epohe (`ExponentialLR`). Osim toga korištena je L2 regularizacija parametara jačine $\lambda = 10^{-8}$.

Model je transformera treniran uz fiksiranje kodere i dekodere te je također treniran stohastičkim gradijentnim spustom uz Adam optimizator. Stopa učenja je $\epsilon = 10^{-3}$ te snaga regularizacije $\lambda = 10^{-10}$. Kao prilagođivač stope učenja ovaj put koristimo kosinusno simulirano kaljenje (`CosineAnnealingWarmRestarts`). Parametri prilagođivača su: $T_0 = 14$, $T_{mult} = 2$ te $\eta_{min} = 10^{-9}$. Trening je proveden u 300 epoha s veličinom minigrupe 32.

Dodatak B: Podatci

Podatci za treniranje su dobiveni integriranjem jednadžbi Gissingerovog sustava (18, 19, 20) za 2048 početnih uvjeta. Vrijeme integracije je 15 jedinica vremena. Skup podataka za testiranje dobiven je istim putem, osim što je integrirano 64 početnih uvjeta, u 75 vremenskih jedinica. U oba su slučaja početni uvjeti uniformno uzorkovani iz volumena $[-2.3, 2.3] \times [-2.3, 2.3] \times [-2.3, 2.3]$. Diskretizacija vremena iznosila je $\Delta t = 0.03$, napominjemo da će i transformer naljediti tu diskretizaciju vremena. Nakon treniranja kodera i dekodera oba su skupa podataka preslikana u latentni prostor dimenzije 32 za treniranje transformera. Ovi su modeli trenirani 200 epoha.

Dodatak C: Dodatni grafovi

U ovom dodatku prikazujemo grafove za tri različita početna uvjeta. Prvi prikazuje tipičnu performansu modela (precizna predviđanja na vremenskoj skali $\sim 2\tau_L$), drugi iznimno lošu performansu modela (previđanja gube preciznost nakon $\sim 0.5\tau_L$) te primjer iznimne performanse (precizna predviđanja na vremenskoj skali $\sim 4.5\tau_L$).

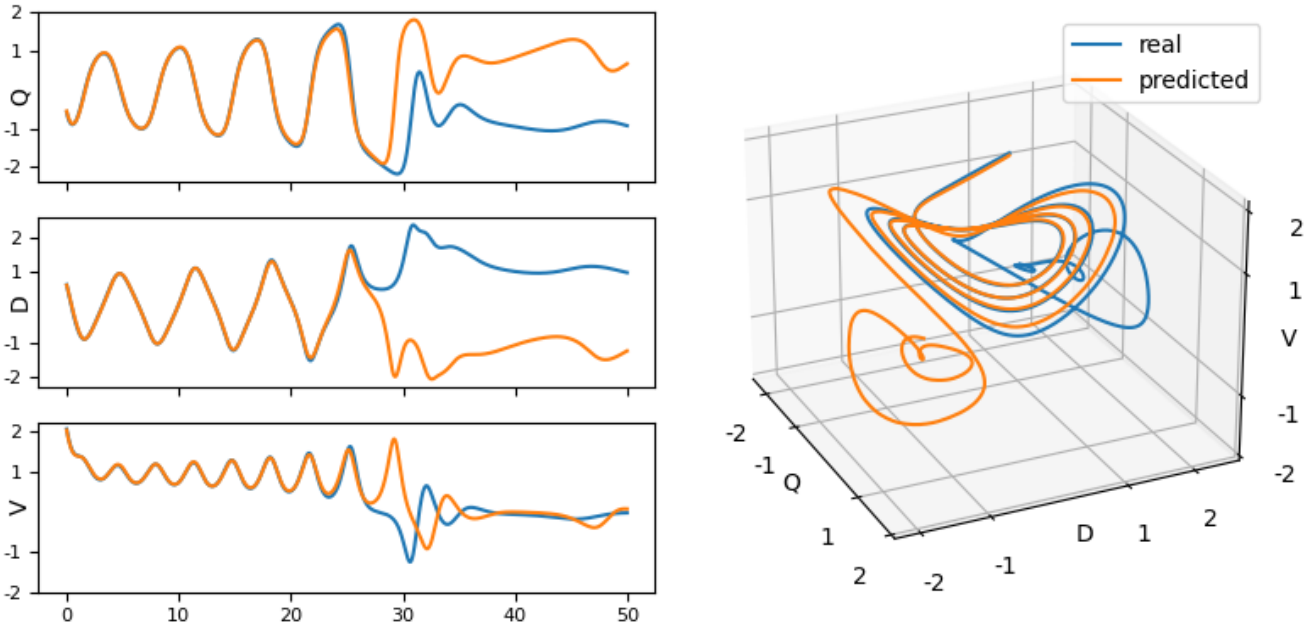
[1] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, USA, 2016) `http://www.deeplearningbook.org`.

[2] N. Geneva and N. Zabaras, Transformers for modeling physical systems, Neural Networks **146**, 272 (2022).

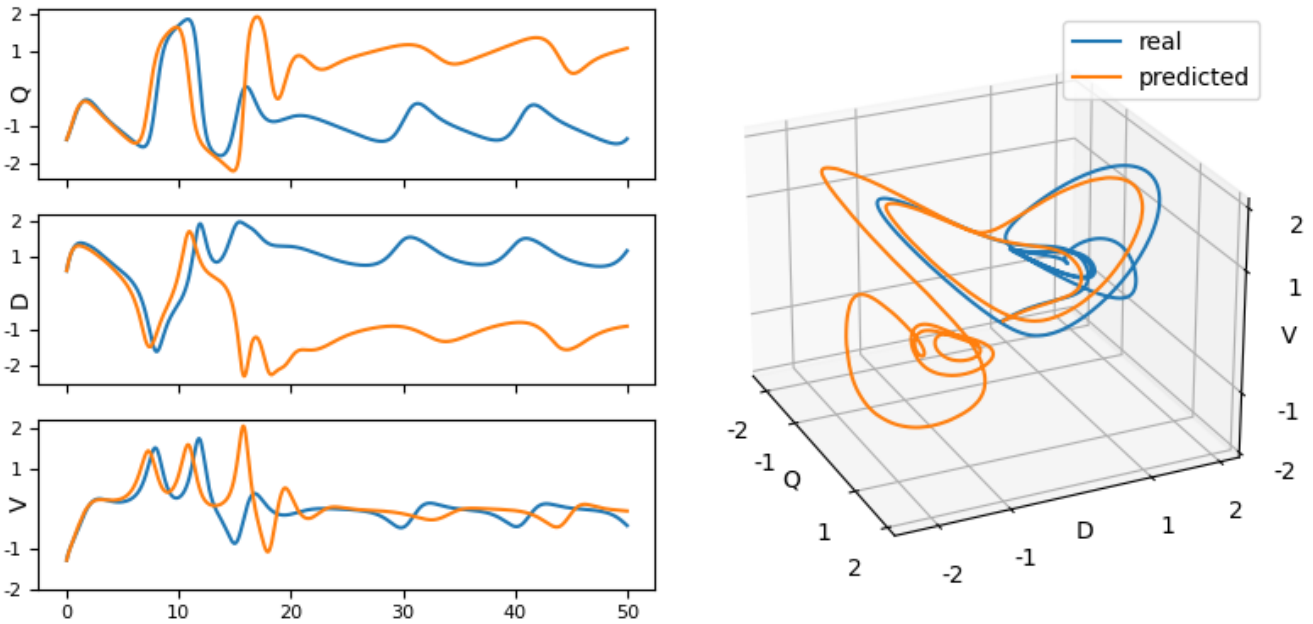
[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, Attention is all you need, in *Advances in Neural Information Processing Systems*, Vol. 30, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc., 2017).

[4] J. S. K. A. Austin Huang, Suraj Subramanian and S. Biderman., The Annotated Transformer, `http://nlp.seas.harvard.edu/annotated-transformer/` (2022).

[5] D. Bahdanau, K. Cho, and Y. Bengio, Neural machine translation by jointly learning to align and translate

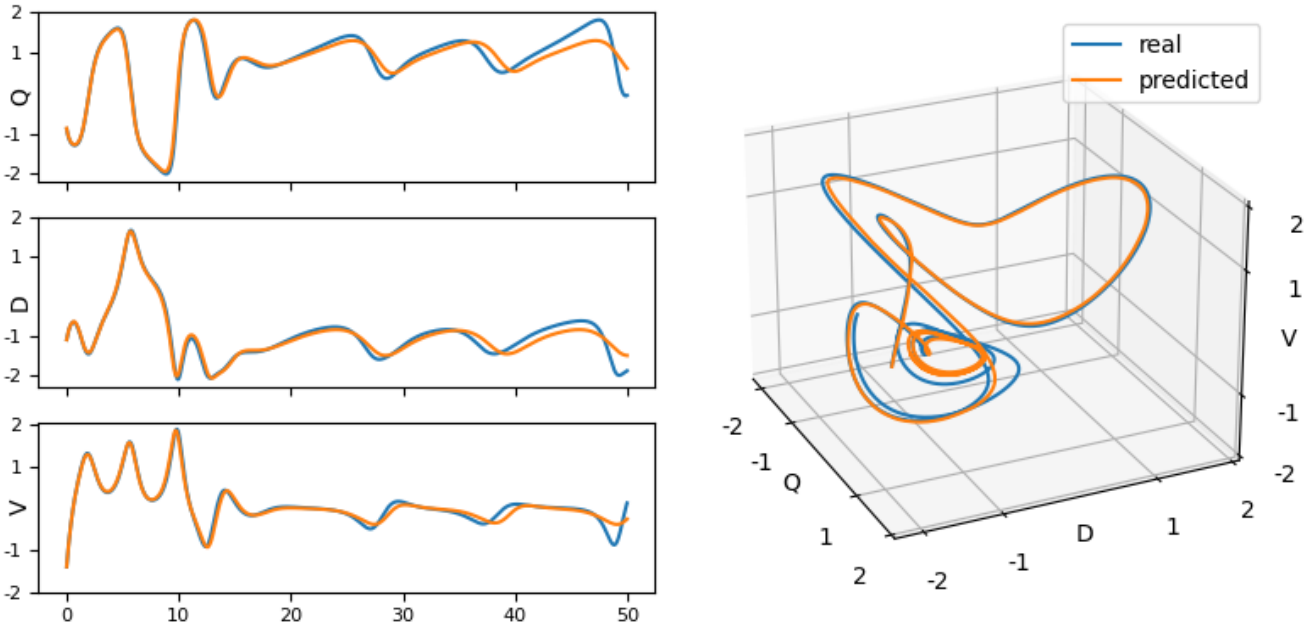


Slika 9. Primjer tipičnog predviđanja modela. Predikcije gube preciznost nakon dva Lyapunovljeva vremena.



Slika 10. U nekim točkama faznog prostora model naglo gubi preciznost u vremenu kraćem od jednog Lyapunovljeva vremena.

(2014).
[6] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, Geometric deep learning: Grids, groups, graphs, geodesics, and gauges (2021).
[7] D. P. Kingma and M. Welling, An introduction to variational autoencoders, Foundations and Trends® in Machine Learning **12**, 307 (2019).
[8] I. Mezic, Koopman operator, geometry, and learning (2020).
[9] A. S. Dogra and W. T. Redman, Optimizing neural networks via koopman operator theory 10.48550/ARXIV.2006.02361 (2020).
[10] D. Luo, J. Shen, R. Dangovski, and M. Soljačić, Koopman operator learning for accelerating quantum optimi-



Slika 11. Primjer izuzetne uspješnosti modela, predviđanja su precizna na jako dugim vremenskim skalama.

zation and machine learning (2022).

[11] P. Bevanda, S. Sosnowski, and S. Hirche, Koopman operator dynamical models: Learning, analysis and control, Annual Reviews in Control **52**, 197 (2021).

[12] P. Mehta, M. Bukov, C.-H. Wang, A. G. Day, C. Richardson, C. K. Fisher, and D. J. Schwab, A high-bias, low-variance introduction to machine learning for physicists, Physics Reports **810**, 1 (2019).

[13] C. Gissinger, A new deterministic model for chaotic reversals, The European Physical Journal B **85**, 10.1140/epjb/e2012-20799-5 (2012).

[14] J. L. Garcia and A. R. Jimenez, Phase space learning with neural networks (2020).

[15] M. Tabor, *Chaos and Integrability in Nonlinear Dynamics-An Introduction* (Wiley, 1989).

[16] T. S. Parker and L. O. Chua, *Practical Numerical Algorithms for Chaotic Systems* (Springer-Verlag, Berlin, Heidelberg, 1989).